

# TOWARDS DESIGN OF SYSTEM HIERARCHY (RESEARCH SURVEY)

Mark Sh. Levin \*

The paper addresses design/building frameworks for some kinds of tree-like and hierarchical structures of systems. The following approaches are examined: (1) expert-based procedure, (2) hierarchical clustering; (3) spanning problems (e.g., minimum spanning tree, minimum Steiner tree, maximum leaf spanning tree problem; (4) design of organizational “optimal” hierarchies; (5) design of multi-layer (e.g., three-layer) k-connected network; (6) modification of hierarchies: (i) modification of tree via condensing of neighbor nodes, (ii) hotlink assignment, (iii) transformation of tree into Steiner tree, (iv) restructuring as modification of an initial structural solution into a solution that is the most close to a goal solution while taking into account a cost of the modification. Combinatorial optimization problems are considered as basic ones (e.g., classification, knapsack problem, multiple choice problem, assignment problem). Some numerical examples illustrate the suggested problems and solving frameworks.

*Keywords:* hierarchy, multi-layer structure, tree, networks, ontology, information systems, solving strategy, heuristics, combinatorial optimization, knapsack, multiple choice problem, spanning tree, Steiner tree, hotlink assignment, multicriteria ranking

## Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>2</b>
<b>2</b>	<b>BASIC HIERARCHIES AND SOME DESIGN APPROACHES</b>	<b>2</b>
2.1	Basic Types of Hierarchies . . . . .	2
2.2	Expert-based 'Top-Down' Procedure . . . . .	4
2.3	Hierarchical Clustering (Agglomerative Algorithm) . . . . .	5
2.4	Towards Ontology . . . . .	7
2.5	Spanning Trees . . . . .	7
2.5.1	Minimum Spanning Tree and Steiner Tree Problems . . . . .	7
2.5.2	Maximum Leaf Spanning Tree Problem . . . . .	9
2.6	Towards Optimal Organizational Hierarchies . . . . .	10
2.7	Multi-layer Structures . . . . .	11
2.7.1	Multi-layer Approach . . . . .	11
2.7.2	Typical Hierarchical Layers in Communication Network . . . . .	11
2.7.3	Layered k-connected Network . . . . .	12
2.7.4	Towards Hierarchical Network Design Problems . . . . .	14
2.7.5	Connection Assignment in Two-layer Network (Access Points - Users) . . . . .	16
2.8	Morphological Hierarchy . . . . .	17
<b>3</b>	<b>SOME APPROACHES TO MODIFICATION</b>	<b>19</b>
3.1	Modification of Tree via Condensing of Weighted Edges . . . . .	19
3.2	Hotlink Assignment Problem . . . . .	23
3.3	Scheme for Transformation of Tree to Steiner Tree . . . . .	25
3.4	Towards Restructuring Problems . . . . .	27
<b>4</b>	<b>CONCLUSION</b>	<b>28</b>

---

\*Mark Sh. Levin: <http://www.mslevin.iitp.ru>; email: [mslevin@acm.org](mailto:mslevin@acm.org)

## 1. INTRODUCTION

Hierarchies play a crucial role as a very useful model for complex systems in all domains: (1) hierarchies leads to a decomposition (i.e., partitioning) of the system representation and corresponding system problems (e.g., design, improvement, maintenance); (2) a hierarchy is often an excellent basis for fast algorithms and/or solving procedures design (i.e., an analogue of linear or convex functions in continuous mathematics); (3) hierarchical approach is the best one for structures of information/knowledge (e.g., ontology); (4) hierarchical approach is the best one for problem solving strategies (e.g., decision trees); (5) hierarchies are very understandable for humans and can be used as a basis of easy learning/teaching interactive procedures. Evidently, many years various methods have been used to design the hierarchical structures (e.g., [1], [4], [21], [41], [62], [75], [78], [83], [84], [87], [94], [95], [98], [105], [115], [118], [151], [155], [159], [161], [168], [202], [208], [214]).

In the article, the following two basic problems are examined:

**I.** Design approaches to design the hierarchies (including expert-based procedure, hierarchical clustering and spanning trees).

**II.** Schemes for transformation of hierarchies (e.g., hotlink assignment, transformation of tree into Steiner tree, restructuring).

Mainly, the considered problems are briefly described via a framework: (i) engineering description, (ii) problem formulation(s), (iii) some problem versions, (iv) basic solving schemes, (v) new prospective problem versions (e.g., multicriteria problems, problems under uncertainty). Numerical examples illustrate the described approaches.

## 2. BASIC HIERARCHIES AND SOME DESIGN APPROACHES

### 2.1. Basic Types of Hierarchies

Generally, it is reasonable to point out some basic types of hierarchies (e.g., [75], [95], [115]):

- (1) various kinds of trees (e.g., Fig. 1, Fig. 2, Fig. 3) (e.g., [71],[75],[115]);
- (2) organic hierarchy (i.e., with organic interconnection among children-vertices, Fig. 4) [39];
- (3) “basic” hierarchy as a tree with additional edges (Fig. 5) (e.g., [122]);
- (4) “morphological hierarchy” (e.g., [123],[124],[128]) (Fig. 6);
- (5) multi-layer structures (e.g., multi-layer networks, hierarchical networks) (Fig. 7) (e.g., [1],[16],[21],[98],[130],[168]).

Here it is reasonable to point out some important research directions in modeling of various multi-layer graphs/networks, for example: (a) hypergraphs (e.g., [17],[18]) and hypernetworks (e.g., [93],[105]); (b) multi-layer social networks (e.g., [109],[152]); (c) multi-stratum networks (e.g., [153]); (d) multi-layer computer systems [196]; (e) multi-layer communications [195]; (f) multi-layer (hierarchical) information-communication networks (e.g., [1],[16],[21],[118],[130],[155],[162],[168],[188]).

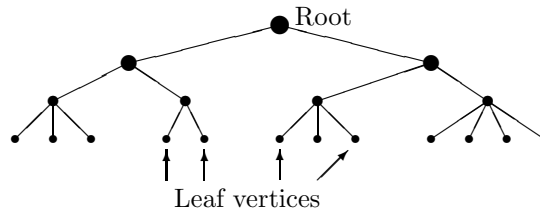


Fig. 1. Tree

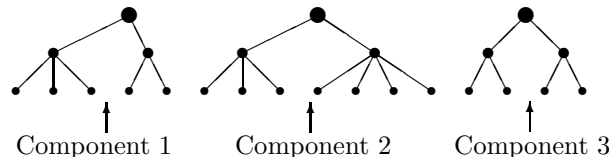


Fig. 2. Forest

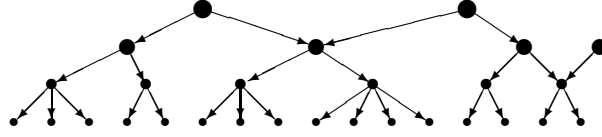


Fig. 3. Polytree

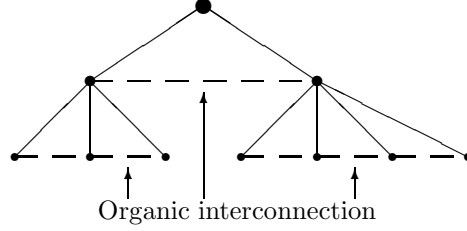


Fig. 4. Organic hierarchy

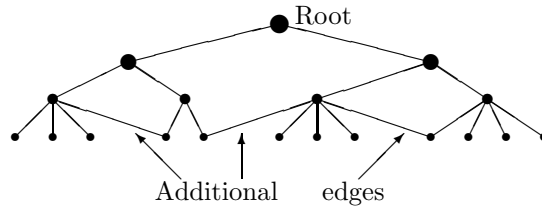


Fig. 5. Hierarchy (tree with additional edges)

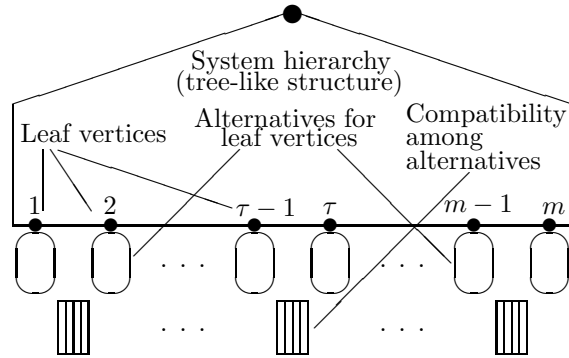


Fig. 6. "Morphological" system hierarchy ([128],[130])

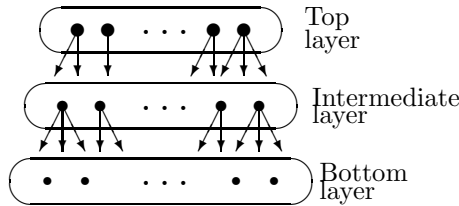


Fig. 7. Multilayer structure

In applied domains, many special types of tree-like structures or hierarchies are widely used, for example: (1) hierarchical schemes for data, for information systems (e.g., [27], [115], [120], [122], [178]); (2) thesauri and concept spaces (e.g., [34],[35]); (3) organizational hierarchies (e.g., [14], [61], [87], [160], [202]); (4) multi-level complex systems (e.g., [158]); (5) phylogenetic trees (e.g., [159],[179],[185]) and evolutionary trees (e.g., [7],[159]); (6) ontologies (e.g., [41], [99],[164],[201]); (7) statecharts (e.g., [23], [97]); (8) decision trees (e.g., [4], [78], [79], [83], [84], [176], [177]); (9) hierarchy of criteria in decision making (Analytic Hierarchy Process) [182]; and (10) hierarchical access networks (e.g., [81]).

Fig. 8 depicts a basic design process to obtain a hierarchical structure.

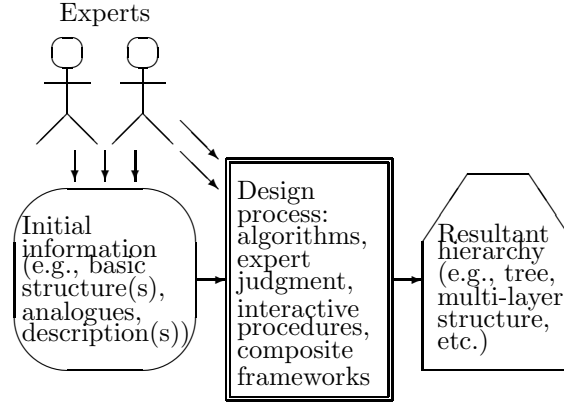


Fig. 8. Building of hierarchy

It is reasonable to list our main types of hierarchy design problems as the following:

**Problem 1.** Expert-based design:

*Input:* description of a system.

*Output:* hierarchy as a result of system partitioning (tree, hierarchy, layered structure).

**Problem 2.** Basic design:

*Input:* set of elements, element attributes or element interconnection.

*Output:* set of clusters or spanning/covering structure over the initial element set (tree, hierarchy, layered structure).

**Problem 3.** Spanning/covering:

*Input:* initial network/graph (i.e., set of elements and set of edges).

*Output:* spanning/covering hierarchical structure (tree, hierarchy, layered structure).

**Problem 4.** Redesign (modification, transformation, improvement):

*Input:* initial hierarchical structure (tree, hierarchy, layered structure).

*Output:* new hierarchical structure with some required features (tree, hierarchy, layered structure).

**Problem 5.** Restructuring (special case of modification):

*Input:* (i) initial hierarchical structure (tree, hierarchy, layered structure), (ii) goal hierarchical structure (tree, hierarchy, layered structure).

*Output:* new hierarchical structure (tree, hierarchy, layered structure) while taking into account the following: (a) “cheap” transformation of the initial structure, (b) “small” proximity between the new structure and the goal structure.

Note, an important problem corresponds to aggregating some several initial hierarchies into a resultant aggregated structure (e.g., aggregation/integration/merging of information systems, data base schemas, catalogs, ontologies, knowledge bases, organizational structures, experts preferences) has been intensively examined (e.g., [3], [15], [34],[40], [48], [145], [165], [166], [174], [207], [213]). An author recent survey on aggregation of some structures is presented in [128].

## 2.2. Expert-based 'Top-Down' Procedure

Mainly, expert-based procedures for building a system hierarchy are based on domain experts and some typical “technological” frames (e.g., product life cycle as the following: design, manufacturing, testing, maintenance, utilization, recycling). This procedure consists of the following phases (it is the ‘divisive’ strategy of hierarchical clustering): 1. dividing a system into its subsystems; 2. dividing each subsystem into its parts; 3. dividing each subsystem part into its components; etc. It is reasonable to point out the basic algorithmic rules for dividing the system (subsystem, subsystem parts, etc.): (a) dividing (partitioning) by physical parts, (b) dividing by system functions, (c) dividing by time stages of data processing. Three applied illustrative examples are presented to illustrate the procedure above:

- (1) for concrete macrotechnology (Fig. 9) ([124], [135]);
- (2) for a two-floor building (Fig. 10) ([124], [137]); and
- (3) for medical treatment (children asthma) (Fig. 11) ([124], [136]).

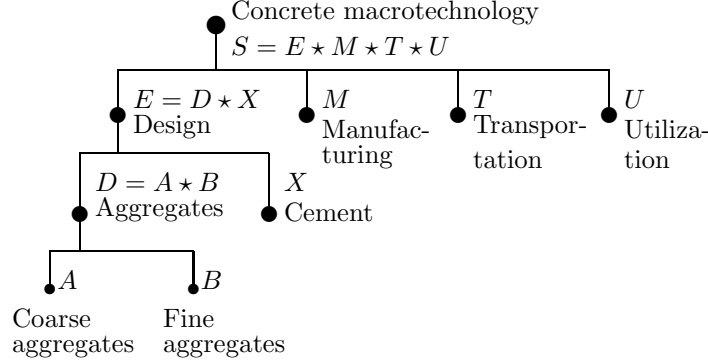


Fig. 9. Hierarchy of concrete macrotechnology ([124],[135])

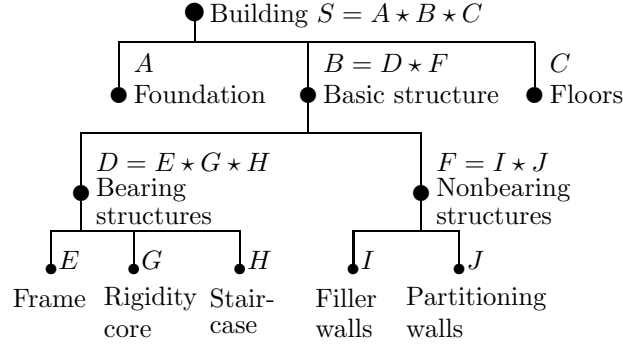


Fig. 10. Hierarchy of building ([124],[137])

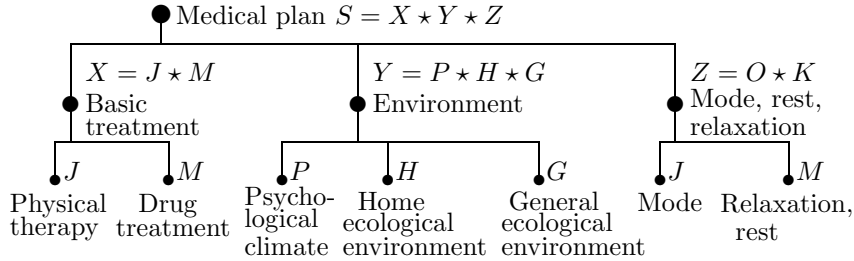


Fig. 11. Hierarchy of medical plan (children asthma) ([124],[136])

### 2.3. Hierarchical Clustering (Agglomerative Algorithm)

Hierarchical clustering consists in building a hierarchy (tree-like structure) of clusters. There is a set of  $n$  elements  $A = \{A_1, \dots, A_i, \dots, A_n\}$  and a corresponding vector estimate of  $m$  attributes/parameters  $(T_1, \dots, T_j, \dots, T_m)$  for each element  $i$ :  $z_i = (z_{i,1}, \dots, z_{i,j}, \dots, z_{i,m})$ . The basic agglomerative algorithm (polynomial, *algorithm 1*) is as follows ('Bottom-Up' element pair integration process) (e.g., [85], [191]):

*Stage 1.* Computing the matrix of element pair  $\forall(A(i_1), A(i_2))$ ,  $A(i_1) \in A$ ,  $A(i_2) \in A$ ,  $i_1 \neq i_2$  "distances" (a simple case, Euclidean distance):

$$d_{i_1 i_2} = \sqrt{\sum_{j=1}^m (z_{i_1, j} - z_{i_2, j})^2}.$$

*Stage 2.* Revelation of the smallest pair "distance" and integration of the corresponding two elements into a resultant "integrated" element.

*Stage 3.* Stopping process or re-computing the matrix of pair "distances" and *Go To Stage 2*.

As result, a tree-like structure for the element pair integration process ('Bottom-Up') is obtained (one element pair integration at each integration step). A basic procedure for aggregation of items (aggregation as average values) is as follows ( $J_{i_1, i_2} = A_{i_1} \& A_{i_2}$ ):  $\forall j \ z_{J_{i_1, i_2}, j} = \frac{z_{i_1, j} + z_{i_2, j}}{2}$ . The item pair aggregation process can be based on other functions (*e.g.*, max, min ). Integration of several items can be considered analogically. An illustrative example corresponds to an analysis of 8 persons by their inclination/interests (Table 1). Here four attributes are used (ordinal scale [0, 5]): (i) inclination for mathematics or logical thinking ( $K_1$ ), (ii) interest to music ( $K_2$ ), (iii) interest to sport ( $K_3$ ), and (iv) interest to trips ( $K_4$ ). The result of the basic hierarchical clustering (as a hierarchical structure for eight-person team) is depicted in Fig. 12.

Table 1. Criteria and estimates

Per-son	$K_1$	$K_2$	$K_3$	$K_4$
$A_1$	0	5	2	3
$A_2$	5	2	3	3
$A_3$	4	3	1	2
$A_4$	4	3	4	2
$A_5$	3	5	3	5
$A_6$	1	5	2	5
$A_7$	3	3	5	5
$A_8$	3	3	4	4

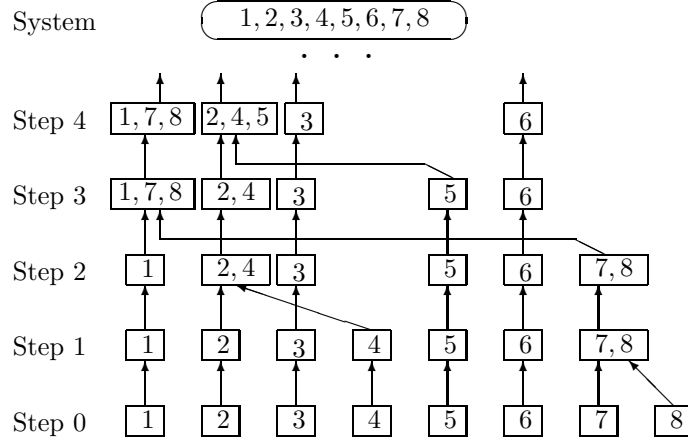


Fig. 12. Example of basic hierarchical clustering

Evidently, modifications of the agglomerative algorithms have been proposed (*e.g.*, [65], [194]). Our approach to modifications of the agglomerative algorithm above is the following [125].

First, computing the item “distance” based on metric  $l_2$  is a very ‘simplified’ mathematical approach. It is possible to examine an ordinal sale for the item “proximity”. As a result, it is possible to select the smallest ordinal item “proximity” and integrate corresponding item pairs.

Second, multicriteria approach for the item “proximity”, (*e.g.*, Pareto approach [169]) can be used.

Third, clustering can be organized as a series revelation of cliques [75].

In addition, it is necessary to note clustering processes can be examined as optimization problems. The basic approach to goal functions in clustering is based on the following: (1) inter-cluster “distances” (*i.e.*, proximity between elements in the same clusters), (2) intra-cluster “distances” (*i.e.*, proximity between elements in different clusters). Some other objectives can be used as well (*e.g.*, number of clusters or closeness to a required interval of cluster numbers, cluster’s cardinalities). Some other objective functions can be used as well (*e.g.*, [100], [106]). Thus multicriteria problem formulations based on criteria above may be examined as well. Generally, it may be very prospective to integrate hierarchical clustering and expert-based interactive procedures (previous section) for the design of system hierarchies.

## 2.4. Towards Ontology

In recent two decades, ontology based approaches have been widely used for representation and processing of information in various domains, for example (e.g., [41],[74], [86], [99],[147],[163],[164],[201]): knowledge-based systems, system design, systems engineering, library science, chemistry, biomedical informatics, conceptual modeling, semantic Web. Ontologies are artifacts as structures (logical, linguistic, “taxonomical”) for description of a domain and/or a “space” of tasks (e.g., [164],[201]). In fact, hierarchical (multi-layer) structures are used here. Basic problems over ontologies are the following (e.g., [41],[99],[164],[165],[166],[201]): (a) design, (b) comparison, (c) integration/merging, (d) alignment. Ontology is often presented as the following activity over various patterns (e.g., logical, reasoning, architectural, naming, content): searching, selecting, composing. A framework for ontology design has been suggested in [164]. An approach to ontology extraction was presented in [73]:

*Stage 1.* Preprocessing (a preliminary work on the available documents is carried out).

*Stage 2.* Creation of the first version of the ontology (as a structure).

*Stage 3.* Creation of concept and relationship. (The creation of the whole ontology extracting the concepts and their relations from the text documents is carried out).

*Stage 4.* Harmonization. The extracted ontology is “harmonized” through the analysis of other domain ontologies and concept description from other systems (e.g., Wikipedia).

*Stage 5.* Refinement and validation. The resultant ontology is refined and validated.

The following main resources for ontologies design are usually pointed out: informal data structures, concept schemes (e.g., classifications, thesauri, nomenclatures), Web-based resources, natural language documents, lexical resources (e.g., dictionaries), modeling languages (e.g., UML, Petri nets). Evidently, modular approaches (based on modular architecture) may be widely used in the ontology design and ontology “life cycle” (i.e., creation, evaluation, testing, utilization, modification).

## 2.5. Spanning Trees

Evidently, spanning tree problems can be used for the design of hierarchical system models, in the case when a preliminary network system model over system elements exists). Many decades, spanning trees problems are used in applications (e.g., network design and maintenance, communication protocol design, VLSI design) and intensively studied in combinatorial optimization (e.g., [42], [59], [75], [82], [101], [115], [210], [211]). In this section, three basic spanning problems (and their modifications) are briefly described: (a) minimum spanning tree problem, (b) minimum Steiner tree problem, and (c) maxim leaf spanning tree problem.

### 2.5.1. Minimum Spanning Tree and Steiner Tree Problems

In this section, a brief description of spanning trees problems is presented (from [126]). Table 2 contains a list of basic spanning tree problems and corresponding literature sources. Fig. 13 illustrates two spanning tree problems.

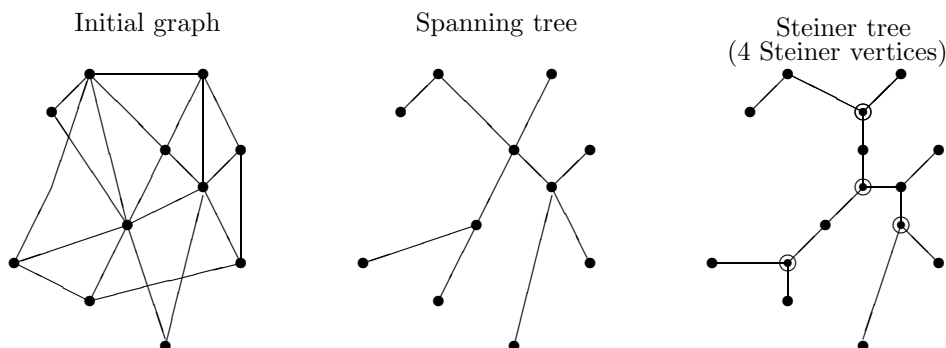


Fig. 13. Illustration for spanning trees [130]

The basic *spanning* problem is the *minimum spanning tree* or the *minimum weight spanning tree* problem (e.g., [42], [57], [72], [75], [172]). Let  $G = (A, E)$  be a connected graph ( $A$  is the set of vertices,  $E$  is the

set of edges/arcs) with nonnegative weights of edges/arcs.

A spanning tree of the graph  $T = (A, E')$  ( $E' \subseteq E$ ) is a subgraph which is a tree and connects all the vertices together. The total weight (cost) of the spanning tree  $c(T)$  is the sum of weights of all its edges/arcs (i.e.,  $E'$ ). A *minimum spanning tree* or *minimum weight spanning tree*  $T^*$  is a spanning tree with the total weight less or equal to the weights of every other spanning tree  $c(T^*) = \min_{\{T\}} c(T)$ . The problem is polynomially solvable (e.g., [42], [72], [75], [172]). The basic well-known polynomial algorithms are the following (e.g., [42], [75]: (a) Prim's algorithm, (b) Kruskal's algorithm, (c) Boruvka's algorithm.

In more general case, a spanning structure corresponds to *spanning forest*: any graph (not necessary connected) has a *minimum spanning forest* which is a union of minimum spanning trees for its connected components. (e.g., [75], [173]).

Table 2. Studies in spanning trees

Spanning problem	Sources
1.Spanning tree	
1.1.Minimum spanning tree	[42],[72],[75],[172]
1.2.Minimum diameter spanning tree	[80]
1.3.Minimum spanning forest	[75],[173]
1.4.Minimum spanning multi-tree	[77],[103],[198]
1.5.Multicriteria spanning tree	[9],[33],[51],[96]
2.Steiner tree	
2.1.Bottleneck Steiner tree problem	[12],[58],[59]
2.2.Steiner tree problem with minimum number of Steiner points	[32],[58],[59],[143]
2.3.Terminal Steiner tree problem	[54],[68],[144]
2.4.Node weighted Steiner tree problem	[89],[113],[189],[216]
2.5.Prize-collecting Steiner problem	[92],[101],[146],[200]
2.6.Survivable Steiner network problem	[59]
2.7.Steiner tree coloring problem	[59]
2.8.Steiner tree scheduling problem	[59]
2.9.Constrained Steiner tree problem	[44],[180]
2.10.Steiner tree problem with hop constraints	[44],[204]
2.11.Steiner tree problems with profits	[43]
2.12.Generalized Steiner problem	[2],[11],[56],[57],[209]
2.13.Generalized Steiner star problem	[112]
2.14.Stochastic Steiner tree problem	[90],[91]
2.15.Dynamic Steiner tree problem	[102]
2.16.On-line Steiner tree problem	[5],[11],[209]
2.17.Group Steiner tree problem	[31],[60]
2.18.Steiner forest problem	[64]
2.19.Multicriteria Steiner tree problem	[140],[141],[206]
2.20.Multicriteria Steiner tree problem with the cost of Steiner vertices	[141]

In recent years, *multicriteria spanning tree* problems (or *multi-objective spanning tree* problems) are examined (e.g., [9],[33], [51], [96]). Here a vector-weight corresponds to each graph edge/arc and the objective function is a vector-like one where each component of the vector above equals a sum of corresponding components of the edges/arcs weights:  $\vec{c}(T)$ . In this case, it is reasonable to search for Pareto-efficient (by  $\vec{c}(T)$ ) solutions. This problem is NP-hard (even in bicriteria case) (e.g., [9],[33]). Here the following algorithms are used (e.g., [9], [33]): (i) multicriteria Prim's spanning tree algorithm, (ii) multicriteria Kruskal's spanning tree algorithm, (iii) genetic algorithms, (iv) multiobjective evolutionary optimization algorithms, (v) heuristics (e.g., GRASP algorithm, simulated annealing algorithm), and (vi) knowledge-based approaches.

In the case of Steiner tree problem (e.g., [2],[11],[56],[57], [59],[75],[82], [101], [113],[189],[209],[210]), the spanning tree can include additional vertices (i.e., Steiner vertices). Thus, the total weight (cost) of the



resultant spanning structure can be less than in the case of the basic spanning tree problem. The basic formulation of *Steiner tree problem* is the following. Given an undirected connected graph  $G = (A, E)$  with node/vertices set  $A$ , edge set  $E$ , and nonnegative weights associated with the edges. Given a set  $Q$  of specified vertices (terminals/basic vertices, i.e., Steiner points). The problem is:

*Find a minimum cost subgraph  $T_s = (Q, E') \subseteq G$  ( $E' \subseteq E$ ) such that there exists a path in the subgraph  $T_s$  between every pair of basic vertices.*

Here the optimal solution  $T_s$  is a tree (Steiner tree). The Steiner tree problem is NP-hard. Descriptions of many Steiner tree problem formulations are presented in [82]. Mainly, the following solving approaches are used for the Steiner tree problems: (1) exact algorithms (enumerative algorithms, for example, branch-and-bound, branch-and-cut) (e.g., [37], [146]) and (2) various heuristics (e.g., [44], [203], [205]) including the following: (a) fast (greedy) algorithms (e.g., [31], [44]); (b) approximation algorithms (e.g., [2], [59], [89], [91], [92], [113], [216]); (c) genetic algorithms (e.g., [108]); (d) AI-based methods (e.g., [107]); (e) dual heuristics (e.g., [49]); (f) distributed primal-dual heuristic (e.g., [187]); and (g) local search techniques (e.g., [29]).

In *multicriteria Steiner tree* problems (e.g., [140], [141], [206]) a vector weight for each edge/arc is under consideration, and the vector-like objective function (as a vector of corresponding sums) can be used (in a simplest case):  $\vec{c}(T_s)$ . Thus, Pareto-efficient solutions (by  $\vec{c}(T_s)$ ) are searched for (e.g., [140], [141]). Here, the following algorithmic approaches can be pointed out: (i) basic heuristics (e.g., [206]); (ii) special multi-layer macro-heuristics: (a) partition-synthesis heuristic (e.g., [123]), (b) spanning-tree based heuristic (e.g., [140]), (c) composite multistage solving scheme (e.g., [141]).

### 2.5.2. Maximum Leaf Spanning Tree Problem

The “maximum leaf spanning tree” problem is the following (e.g., [6], [75], [114]):

*Find a spanning tree of an input graph so that the number of the tree leafs is maximal.*

Generally, the spanning tree of a graph contains the following types of nodes: (a) root, (b) internal nodes (the internal nodes may be considered as a virtual “bus” in networking), and (c) leaf nodes. Thus, the problem consists in maximizing the number of leaf nodes or minimizing the number of internal nodes. The problem is one of the basic NP-hard problems [75]. Fig. 14 depicts an illustrative numerical example.

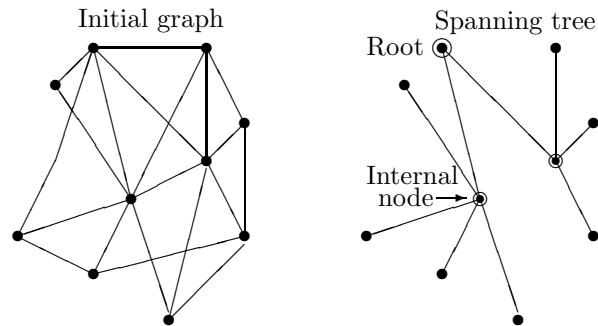


Fig. 14. Illustration for maximum leaf spanning tree

The problem is applied in networking (e.g., minimum-energy broadcast trees in ad hoc wireless networks), (e.g., [142], [199]), in circuit layout [193]. The following main types of algorithms have been suggested: (1) exact algorithms (branch-and bound) ([66], [70]), (2) local optimization [148], (3) greedy 3-approximation algorithm (linear time) [149], (4) 2-approximation algorithm [192], and (5) heuristics (e.g., Bee Colony algorithm) (e.g., [55], [190]). Some prospective generalizations of the “maximum leaf spanning tree” problem may be considered while taking into account the following: (a) multicriteria descriptions of leaf nodes, (b) uncertainty, (c) dynamics.

In sense of exact algorithms, this problem is equivalent to “connected dominating set” problem (e.g., [22], [30], [36], [75]):

Find a minimum set of vertices  $D \subseteq A$  of input graph  $G = (A, E)$  that the induced by  $D$  subgraph  $G' = (D, E')$  ( $E' \subseteq E$ ) is connected dominated set and  $D$  is a dominating set of  $G$ .

A recent survey on the problem is presented in [22]. The “connected dominating set” problem plays the central role in sensor wireless networks, in mobile ad-hoc networks (MANETs), in network testing (e.g., [22], [47], [142], [199]). This problem is used in communication protocols including the following [22]: (i) media access coordination, (ii) unicast, (iii) multicast/broadcast, (iv) location-based routing, (v) energy conservation, (vi) topology control, and (vii) resource discovery in MANET. Mainly, the following algorithms are used for the problem: (i) approximation algorithms (e.g., [36], [88]), (ii) heuristics (e.g., [28], [186]). Analogically, prospective generalizations of the problem may involve the following: (a) multicriteria descriptions of the elements of the dominating set, (b) uncertainty, (c) dynamics.

## 2.6. Towards Optimal Organizational Hierarchies

Organizational hierarchies have been investigated many decades (e.g., [14], [50], [87], [154], [160], [197], [202], [208]). The following basic approaches to organizational structures are used in firms (manufacturing, sales, etc.): bureaucratic structures, functional structures, division structure (or product structure), matrix structure (integration of functions and products). Generally, the following hierarchical layers are considered for organizations: (a) organization, (b) branches, (c) departments (divisions), (d) work groups, (e) individuals. Fig. 15 depicts an example of hierarchical structure for universities.

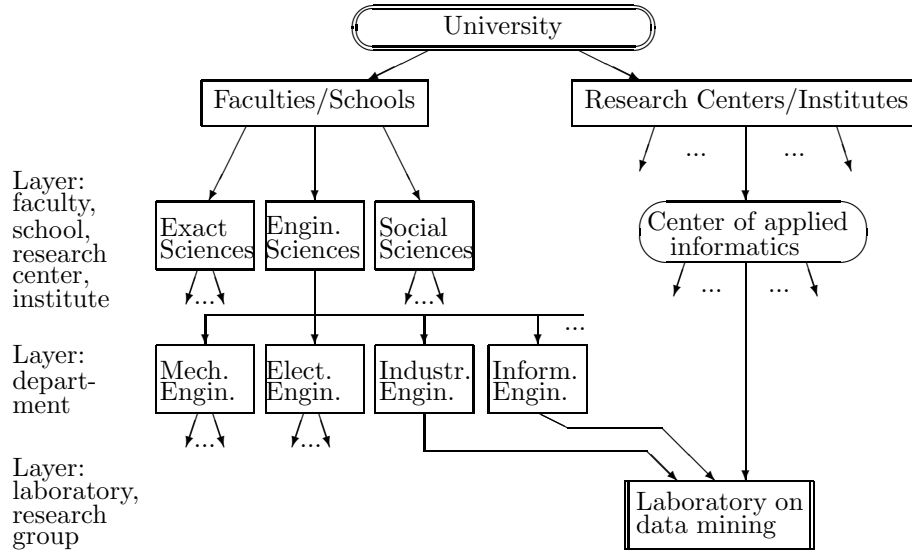


Fig. 15. Example of hierarchical structure (university)

In recent years, interests to design of ‘optimal’ organizational hierarchy have been increased (e.g., [87], [160], [202]). Here, approaches can be based on spanning structures (see previous section) or direct design. Two basic kinds of direct design problems may be examined (e.g., [122], [126]):

1. to build a hierarchy that has the best value(s) of its property(ies);
2. to build a hierarchy that is the most ‘close’ to an ideal one (or a set of ideal structures).

Let  $\{H_i = (A_i, E_i), i = \overline{1, n}\}$  be a set of initial structures (hierarchies). Let  $H^g(A_g, E_g)$  ( $H^g \in \{G_i\}$ ) be a goal (i.e., ideal) structure, let  $\rho(H_{i_1}, H_{i_2})$  be a proximity (or ‘distance’) between two structures (hierarchies)  $H_{i_1}$  and  $H_{i_2}$  ( $\forall H_{i_1}, G_{i_2} \in \{H_i\}$ ). In addition, it is reasonable to consider properties of a structure (e.g., vertex degree, vertex connectivity)  $H_{i_1} \in \{H_i\}$ :  $\overline{\psi(H_{i_1})} = (\psi_1(H_{i_1}), \dots, \psi_\mu(H_{i_1}), \dots, \psi_m(H_{i_1}))$ . Here hierarchy  $H^* \in \{H_i\}$  is searched for. Thus, in the 1st kind of the problem the objective function is (a case of maximization for each property):

$$\max_{H^* \in \{H_i\}} \psi_\mu(H^*) \quad \forall \mu = \overline{1, m}.$$

This problem is close to the well-known class of “graph augmentation” problems, i.e., modification of a graph to get some required properties of the modified graph (e.g., [63],[111]).

In the second kind of the problem the objective function is:  $\min_{H^* \in H_i} \rho(H^*, H^g)$ . In general, multicriteria problem formulations may be examined as well. Clearly, it is possible to consider integrated problem formulations, for example:

$$\min_{H^* \in H_i} \rho(H^*, H^g) \quad s.t. \quad \psi_\mu(H^*) \geq d^\mu \quad \forall \mu = \overline{1, m},$$

where  $(d^1, \dots, d^\mu, \dots, d^m)$  is a vector constraint for properties. The optimization models above usually correspond to complicated integer (or mixed integer) optimization problems and here various solving approaches are used (e.g., enumerative methods, heuristics, AI techniques).

In addition, it is necessary to point out that an attempt to build generalized approach for ‘optimal’ organization hierarchy is presented in ([87], [160], [202]). This approach is based on the usage of a functional  $P$  of hierarchical structure  $G \in \Omega$  ( $\Omega$  is the set of hierarchical structures as direct acyclic ‘layered’ graphs;  $P : \Omega \rightarrow [0, +\infty)$ ) and problem is:  $\arg \min_{G \in \Omega} P(G)$ . Mainly, structure  $G$  is considered as tree and functional  $P$  is considered as convex ([87], [160], [202]). The set of applications involves technological process, supply chain network, etc.

## 2.7. Multi-layer Structures

### 2.7.1. Multi-layer Approach

Multi-layer (or multi-level) approach is a basic one for representation of complex systems (e.g., [130],[152], [158],[195],[196]). In fact, this approach is a basic methodological method for decreasing the system complexity (i.e., multi-layer partitioning a systems and corresponding partitioning the system problems set). Here, the following main methodological steps can be pointed out. First, levels for main properties of complex systems have to be considered, for example: (i) stability, (ii) controlability, (iii) adaptability, and (iv) self-organization [158]. Second, seven-layer structure for computer systems was suggested as follows (e.g., [196]): (1) layer of hardware, (2) layer of microprogramming, (3) layer of operation system, (4) assembler-based layer, (5) layer of algorithmic languages, (6) layer of applied support systems (e.g., DBMS, DSS), and (7) layer of applications. Third, seven basic layers (OSI model) for data transmission in communication networks were suggested (e.g., [195],[215]): (1) physical layer (media, signals and binary transmission), (2) data link layer (physical addressing), (3) network layer (path determination and logical addressing), (4) transport layer (end-to-end connections, reliability and flow control), (5) session layer (interhost communication, managing sessions between applications), (6) presentation layer (data presentation, encryption and decryption, convert machine dependent data to machine independent data), and (7) application layer (network process to application).

Evidently, the multi-layer approach can be applied in many domains. For example, four-layer structure was suggested for representation of combinatorial “optimization problems domain” in [130]: (i) layer of basic combinatorial optimization problems, (ii) layer of multicriteria combinatorial problems, (iii) layer of typical composite problem frameworks, and (iv) layer of typical applications.

In general, a multi-layer system hierarchy consists of the following: (a) hierarchical layers; (b) set of elements for each hierarchical layer, description of the elements (i.e., attributes); (c) interconnections (some relations) over the set of elements for each hierarchical layer; (d) connections between elements of neighbor hierarchical layers. Thus, the following design framework for a multi-layer system hierarchy can be considered:

1. Generation of multi-layer structure (i.e., the layer), for example: (i) dividing the initial elements/nodes into parts corresponding to layers (levels), (ii) description of layer elements (nodes, arcs) (e.g., traffic), (iii) building a structure for each part, e.g., path, multiple paths, tree, ring, complete graph (i.e., clique) or their combinations.
2. Definition (searching for) connections between elements of neighbor hierarchical layers.

### 2.7.2. Typical Hierarchical Layers in Communication Network

First, it is reasonable to describe a typical hierarchy of communication networks (e.g., [45], [76], [118], [162]). A traditional network hierarchy consists of the following basic layers: (a) international (multi-country, continent) network GAN; (b) metropolitan network MN; (c) wide area network WAN; and (d) local area network LAN. *IBM Red Book* contains an interesting dimensional classes of networks by node numbers as follows: (i) large size communication network ( $> 500$  nodes); (ii) medium size

communication network ( $< 500$  nodes); and (iii) small size communication network ( $< 80$  nodes) [162]. From the “engineering” viewpoint, the following hierarchical layers can be considered:

1. Backbone network.
2. Global network as a set of interconnected network segments including the following: (a) additional centers, (b) cross-connections, and (c) bridges.
3. Access network / network segment (cluster): (e.g., bi-connected topology, about 20 nodes).
4. Distributed network: a simple hard topology (e.g., bus, star, tree, ring).

As a result, a class of small-dimensional networks is added to the above-mentioned classification: 20...25 nodes. Here a simplified network hierarchy is examined as follows (Fig 16):

1. *TOP LAYER*: nodes and links for connection of clusters: 1.1. basic node clusters (network segments), 1.2. communication centers, 1.3. cross-connection links (center’s connection), and 1.4. links for connection of neighbor clusters.
2. *MEDIUM LAYER*: basic clusters (network segment/ access networks, bi-connected topological modules).
3. *BOTTOM LAYER*: distribution networks (e.g., tree, ring, bus).

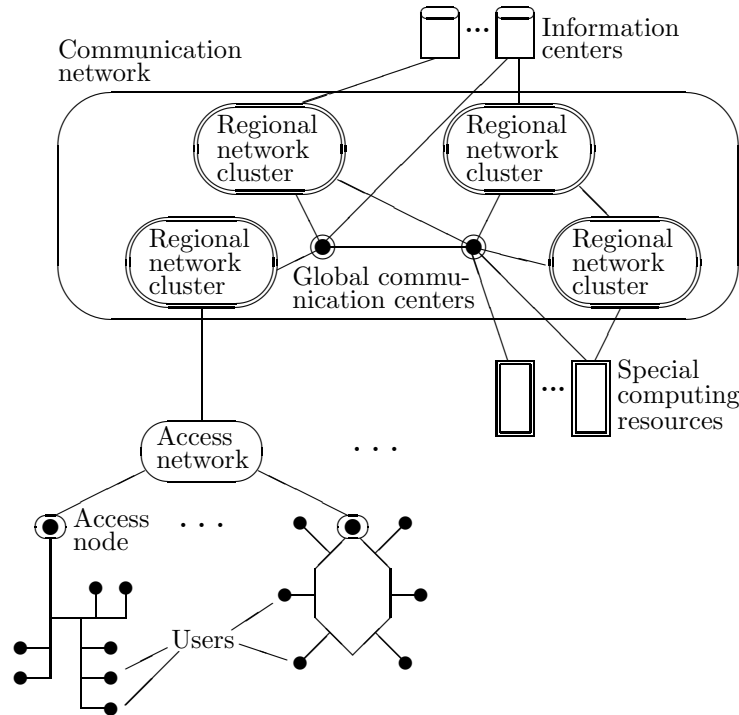


Fig. 16. Illustration for multi-layer communication network

### 2.7.3. Layered $k$ -connected Network

A special version of 2-connected network was suggested in [16]. The generalized  $k$ -connected network of this kind and its design scheme were briefly described in [130]. Let  $G = (A, E)$  be an initial graph (network) where  $A$  is a vertex (node) set,  $E$  is an edge set and  $|A| \geq k \times (k + 1) + k$ .

Our kind of  $k$ -connected networks is: (a)  $k$  “centers” where each “center” is  $(k + 1)$ -vertex clique (the “centers” have not intersections), (b) node set  $M \subseteq A$  ( $|M| \geq k$ ) includes all nodes which do not belong to the “centers”; there is a connection (i.e., edge) between  $\forall \alpha \in M$  and each “center”, i.e.,  $k$  edges (one edge for a connection to a “center”). As a result, the following three-layer network is obtained (Fig. 17): (i) layer of end nodes, (ii) layer of special “central” communication nodes: selected nodes and/or special additional nodes (the nodes are useful for location of key communication equipment); and (iii) communication “center” (the “center” can correspond to “communication providers” / “communication operators” or their branches).

Fig. 18 depicts an example of 4-connected structure. Proof of  $k$ -connectivity for the defined kind of networks is based on four special cases. Consider two nodes  $a, b \in A$ . The cases are as follows:

*Case 1:*  $a$  and  $b$  belong to the same “center” (i.e.,  $(k + 1)$ -clique). Nodes  $a$  and  $b$  have connections as follows: (i) direct connection  $(a, b)$ , (ii)  $(k - 1)$  two-edge connection by other nodes of this “center”. Thus, nodes  $a$  and  $b$  have  $k$  connections (without intersection).

*Case 2:*  $a$  belongs to “center”  $A$  and  $b$  belongs to “center”  $B$  (i.e., another one). Nodes  $a$  and  $b$  have connections as follows: two-edge connection by another node  $\alpha \in M$  (i.e.,  $(k)$ -connections of this kind). Thus, nodes  $a$  and  $b$  have  $k$  connections (without intersection).

*Case 3:*  $a$  belongs to a “center” and  $b \in B$ . Nodes  $a$  and  $b$  have connections as follows: (i) direct connection  $(a, b)$ , (ii)  $(k - 1)$  three-edge connection: by a node of each other “center” and by another node from  $M$  ( $(k - 1)$  different ways). Thus, nodes  $a$  and  $b$  have  $k$  connections (without intersection).

*Case 4:*  $a, b \in M$ . Nodes  $a$  and  $b$  have  $k$  different connections as follows: node  $a$ , a “center”, node  $b$ .

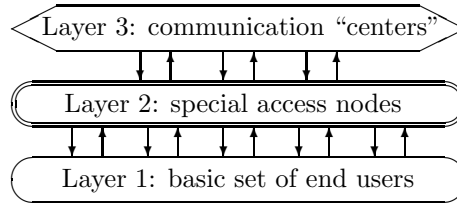


Fig. 17. Three-layer model of network

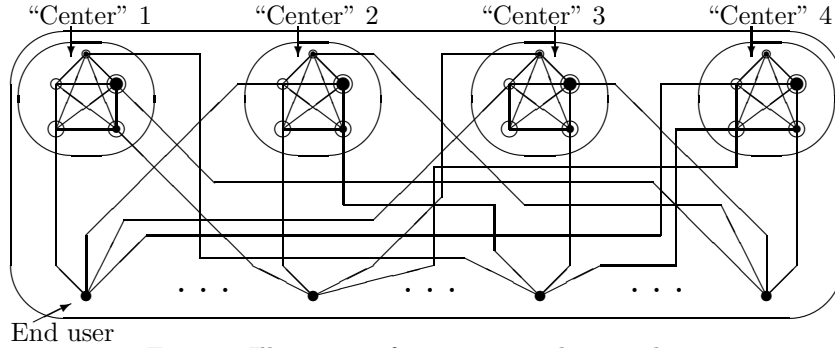


Fig. 18. Illustration for 4-connected network

Further, a solving scheme ('Bottom-Up') to build  $k$ -connected structure is:

*Stage 1.* Selection of  $k \times (k + 1)$  vertices for  $k$  “centers” (here multicriteria ranking problem can be used).

*Stage 2.* Clustering of the selected vertices to get  $k$  clusters as “centers” (each cluster consists of  $(k + 1)$  vertices, the clusters have not intersections).

*Stage 3.* Building of connections for end users: connection of each vertex (that does not belong to a “center”) with each “centers” (i.e., with the only one node of each “centers”). Here multiple choice problem or its modifications can be used.

Now an illustrative example of illustrate the examined type of bi-connected communication network is presented. Initial nodes are depicted in Fig. 19.

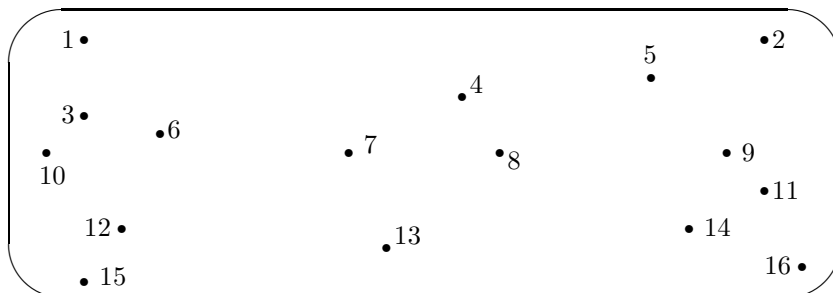


Fig. 19. Initial network nodes

Here the following nodes are selected for “centers”:  $\{3, 5, 9, 10, 12, 14\}$ . Other nodes correspond to users.

The corresponding version (*version 1*) of the resultant bi-connected networks is presented in Fig. 20:

*Version 1.* “Centers” are locally-allocated (i.e., regional “centers”): “center” 1: nodes 3, 10, and 12; “center” 2: nodes 5, 9, and 14.

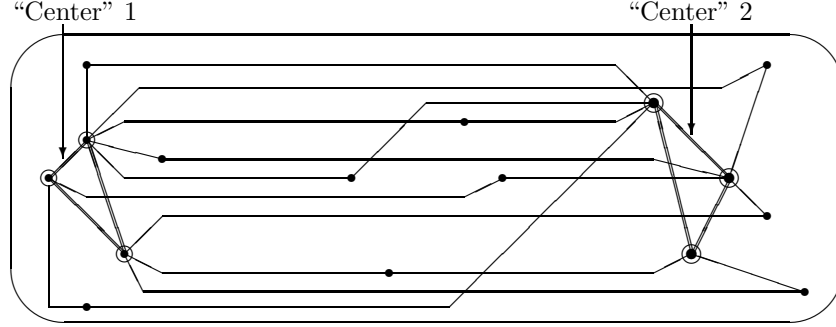


Fig. 20. Bi-connected network: locally-allocated “centers”

On the other hand, it may be reasonable to consider another situation with “simplification” of connections for end users (e.g., minimization of connection distances). In this case, it is necessary to use the following algorithmic rule at *stage 2* of the above-mentioned solving scheme (i.e., grouping of the selected nodes to centers):

*Nodes of each “center” have to ‘cover’ the network (e.g., to be very close to all nodes of end users).*

Thus, the second solving scheme to build  $k$ -connected structure is:

*Stage 1.* Clustering of the initial set of nodes to get  $k$  clusters.

*Stage 2.* Building of  $k$  “centers”: selection of a vertex in each obtained cluster as an element for each “center” (a representative of the “center”), connection of the vertices in each “center”.

*Stage 3.* Building of connections for end users: connection of each vertex (that does not belong to a “center”) with each “centers” (i.e., with the only one node of each “centers”).

Fig. 21 depicts an example of this type of the resultant bi-connected network:

*Version 2.* “Centers” are distributed over the network: “center” 1: nodes 3, 9, and 12; “center” 2: nodes 5, 10, and 14.

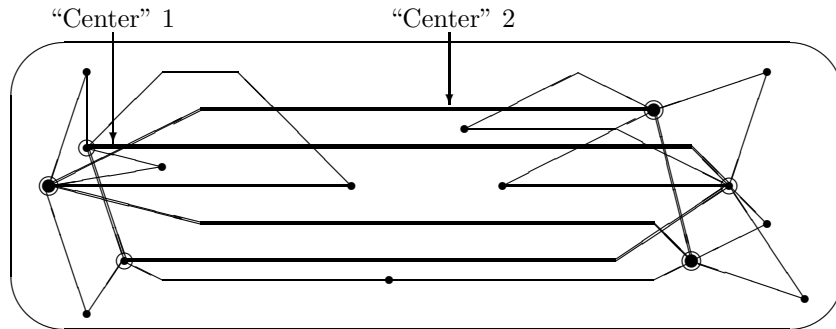


Fig. 21. Bi-connected network: distributed over network “centers”

#### 2.7.4. Towards Hierarchical Network Design Problems

The basic hierarchical two-level network design problem consists in finding a minimum cost two-level spanning network, consisting of two parts: (i) main path (or several paths, tree, ring) (ii) secondary trees (e.g., [45], [167], [168], [175]). Thus, the initial network is divided into two parts:

1. The higher level part (primary nodes, main path): a path (or several paths, tree, ring) composed of primary arcs, which visits some of the nodes of the network (i.e., primary nodes).

2. The lower level part (secondary nodes, secondary trees): the part is composed of one or more trees whose arcs, termed secondary, are less expensive to build than the primary arcs.

Here, each arc has a cost ( $d_{ij}$ ,  $\forall i, j \in A$ ,  $A$  is the set of nodes). The total cost of the selected arcs in the spanning structure is used as the minimized objective function. The problem is formulated as combinatorial optimization model (e.g., [45]), it is NP-hard [13]. Various approaches have been suggested for the problem, for example: (a) exact (enumerative) methods (e.g., branch-and-cut approach, dynamic programming), (b) heuristics (e.g., Lagrangian relaxation), (c) evolutionary algorithms. Often, a preliminary minimum spanning tree is used to construct the solution (leaf nodes of a spanning tree are used as the secondary nodes). Further, three illustrative examples are presented:

(i) higher part is path over primary nodes  $\langle 1, 6, 5, 7, 2, 3, 4 \rangle$  (Fig. 22),

(ii) higher part is tree over primary nodes  $\{1, 2, 3, 4, 5, 6, 7\}$  (Fig. 23),

(iii) higher part is ring over primary nodes  $\{1, 2, 3, 4, 5, 6, 7\}$  (two-connected case) (Fig. 24).

Main applications of the problem involve communication networks, computer networks, transportation networks, power line distributed systems. Evidently, more general multi-level networks are examined as well (e.g., [13], [38]).

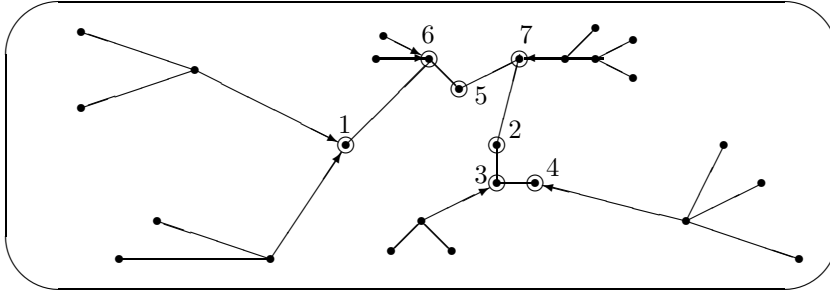


Fig. 22. Example of two-level network (higher level: path)

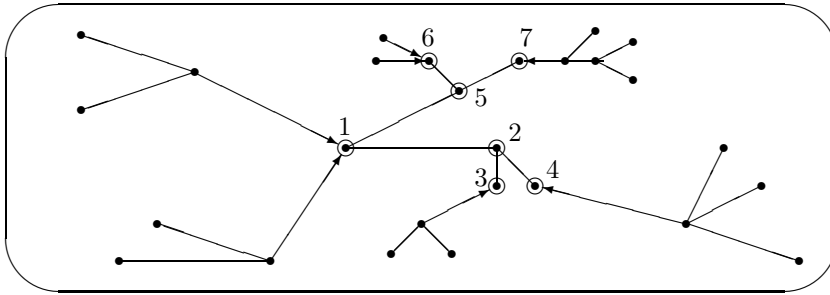


Fig. 23. Example of two-level network (higher level: tree)

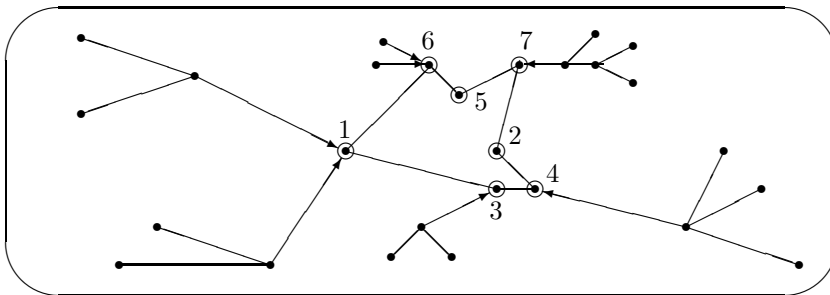


Fig. 24. Example of two-level network (higher level: ring)

### 2.7.5. Connection Assignment in Two-layer Network (Access Points - Users)

This section <sup>2</sup> focuses on assignment design for two-layer network (access points - users) (Fig. 25). The design problem consists in connection (assignment) of each user to access point.

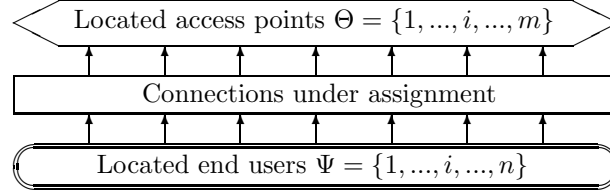


Fig. 25. Layers: access points - users

The problem and initial data are partially based on [138]. The following parameters are used: set of users  $\Psi = \{1, \dots, i, \dots, n\}$  ( $n = 25$ ), set of access points  $\Theta = \{1, \dots, i, \dots, m\}$  ( $m = 6$ ). Each user is described by parameter vector  $(x_i, y_i, z_i, f_i, p_i)$ , where vector components are as follows (Table 3): coordinates of user  $(x_i, y_i, z_i)$ , required frequency bandwidth  $f_i$  (scale: 1 Mbit/s ... 10 Mbit/s), priority  $p_i$  (ordinal scale [1,2,3], all user requirements are satisfied in case  $p_i = 1$ ), required reliability  $r_i$  (ordinal scale [1,10], 10 corresponds to maximum reliability). Analogically, parameters of access points are considered (by index  $j$ , Table 4) including parameter  $n_j$  (maximal possible number of users under service).

Table 3. Data on users [127]

$i$	$x_i$	$y_i$	$z_i$	$f_i$	$p_i$	$r_i$
1	30	165	5	10	2	5
2	58	174	5	5	1	9
3	95	156	0	6	1	6
4	52	134	5	6	1	8
5	85	134	3	6	1	7
6	27	109	7	8	3	5
7	55	105	2	7	2	10
8	98	89	3	10	1	10
9	25	65	2	7	3	5
10	52	81	1	10	1	8
11	65	25	7	6	2	9
12	93	39	1	10	1	10
13	172	26	2	10	2	7

$i$	$x_i$	$y_i$	$z_i$	$f_i$	$p_i$	$r_i$
14	110	169	5	7	2	5
15	145	181	3	5	2	4
16	170	161	5	7	2	4
17	120	140	6	4	2	6
18	150	136	3	6	2	7
19	175	125	1	8	3	5
20	183	91	4	4	3	5
21	135	59	4	13	3	4
22	147	79	5	7	3	16
23	172	26	2	10	2	7
24	165	50	3	7	3	3
25	127	95	5	7	2	5

Table 4. Data on access points [127]

$j$	$x_j$	$y_j$	$z_j$	$f_j$	$n_j$	$r_j$
1	50	157	10	30	4	10
2	72	102	10	42	6	10
3	45	52	10	45	10	10

$j$	$x_j$	$y_j$	$z_j$	$f_j$	$n_j$	$r_j$
4	150	165	10	30	5	15
5	140	112	10	32	5	8
6	147	47	10	30	5	15

Further, each pair “user-access point” (i.e.,  $(i, j), i \in \Psi, j \in \Theta$ ) can be described: (1) reliability  $r_{ij} = \min\{r_i, r_j\}$ , (2) distance  $l_{ij}$ , (3) priority  $p_{ij} = p_i$ , and (4) required bandwidth  $f_{ij} = f_i$ . In addition, a “connection” parameter is considered:  $\beta_{ij}$  equals 1 if  $l_{ij} \leq l$  and 0 otherwise ( $L$  corresponds to distance constraint). This parameter defines  $\forall i \in \Psi$  a subset of possible access points  $\Theta_i \subseteq \Theta$ . The assignment of user  $i$  to access point  $j$  is defined by Boolean variable  $x_{ij}$  ( $x_{ij} = 1$  in the case of assignment

<sup>2</sup> From (with amendments):

M.Sh. Levin, Towards communication network development (structural systems issues, combinatorial problems). *IEEE Region 8 Int. Conf. Sibircon 2010*, vol. 1, (2010) 204-208.



$i$  to  $j$  and  $x_{ij} = 0$  otherwise). The assignment solution  $(\Psi \Rightarrow \Theta)$  is defined by Boolean matrix  $X = ||x_{ij}||$ ,  $i = \overline{1, n}$ ,  $j = \overline{1, m}$ . The problem formulation is:

$$\begin{aligned} & \max \sum_{i=1}^n \sum_{j \in \Theta_i} r_{ij} x_{ij}, \quad \max \sum_{i=1}^n \sum_{j \in \Theta_i} f_{ij} x_{ij}, \quad \max \sum_{i=1}^n \sum_{j \in \Theta_i} p_{ij} x_{ij} \\ & s.t. \quad \sum_{i=1}^n f_{ij} x_{ij} \leq f_j \quad \forall j \in \Theta, \quad \sum_{i=1}^n x_{ij} \leq n_j \quad \forall j \in \Theta, \quad \sum_{j \in \Theta_i} x_{ij} \leq 1 \quad \forall i \in \Psi, \\ & x_{ij} \in \{0, 1\}, \quad \forall i = \overline{1, n}, \quad \forall j = \overline{1, m}; \quad x_{ij} = 0, \quad \forall i = \overline{1, n}, \quad j \in \{\Theta \setminus \Theta_i\}. \end{aligned}$$

This multiple criteria assignment problem is NP-hard (e.g., [75]). Here, a simplified two-stage heuristic is used: (i) transformation of vector estimate for each pair  $(i, j)$  into an ordinal estimate (by multicriteria ranking, ELECTRE-like technique ([134], [181])), (ii) solving the obtained one-criterion assignment problem (by greedy algorithm). Fig. 26 depicts the obtained solution: assignment of users to access points.

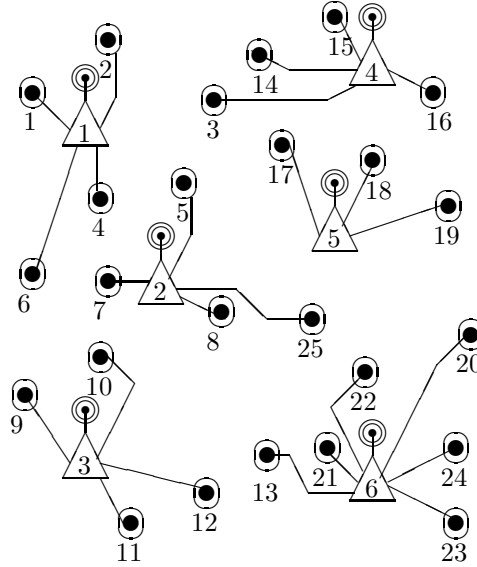


Fig. 26. Assignment: users - access points [127]

## 2.8. Morphological Hierarchy

Morphological hierarchy generalizes system model from morphological analysis [217] by the following ways ([123],[124],[126],[128],[131],[133]): (i) hierarchical (tree-like) structure of the examined system; (ii) design alternative set for each leaf node (system component) of the system model; (iii) assessment of the design alternatives (DAs) on the basis of an ordinal scale (e.g., [123],[124],[126]) or a special interval multiset based scale [133]; and (iv) ordinal compatibility between the design alternatives for different system components (instead of binary compatibility that was used in morphological analysis).

An example of morphological hierarchy for a compressed plan of car repair from [132] is the following (Fig. 27, ordinal estimates of DAs are shown in parentheses; Table 5, Table 6):

0. Plan  $S = A * B * C$

1. Payment  $A = X * F$

1.1. payment scheme  $X$ : 100 % payment  $X_0$ , prepayment of 50...80 percent for parts  $X_1$ ; bank loan  $X_2$ ;

1.2. version  $F$ : cash  $F_1$ , credit card  $F_2$ , bank transfer  $F_3$ .

2. Body  $B = R * Z * M$ :

2.1. frame  $R$ : None  $R_0$ , technical diagnostics  $R_1$ , follow-up assembly  $R_2$ ;

2.2. hardware  $Z$ : None  $Z_0$ , replacement of defect parts  $Z_1$ , repair of body-defects  $Z_2$ , fitting  $Z_3$ ,  
 $Z_4 = Z_1 \& Z_2$ ,  $Z_5 = Z_1 \& Z_3$ ,  $Z_6 = Z_2 \& Z_3$ ,  $Z_7 = Z_1 \& Z_2 \& Z_3$ ;

2.3. finishing  $M = U * V$ :

2.3.1. painting  $U$ : None  $U_0$ , partial painting  $U_1$ , painting  $U_2$ ;

2.3.2. appearance restoration  $V$ : None  $V_0$ , Yes  $V_1$ .

3. Electric & electronic subsystem  $C = P * Q$ :

3.1. Computer & navigation subsystem  $P = K * G$

3.1.1. Computer  $K$ : None  $K_0$ , upgrade  $K_1$ , additional or new computer  $K_2$ ;

3.1.2. system GPS  $G$ : None  $G_0$ , GPS system  $G_1$ ;

3.2. wiring & lighting  $Q = O * L$

3.2.1. wiring  $O$ : None  $O_0$ , repair  $O_1$ ;

3.2.2. lighting  $L$ : None  $L_0$ , partial replacement  $L_1$ , replacement  $L_2$ .

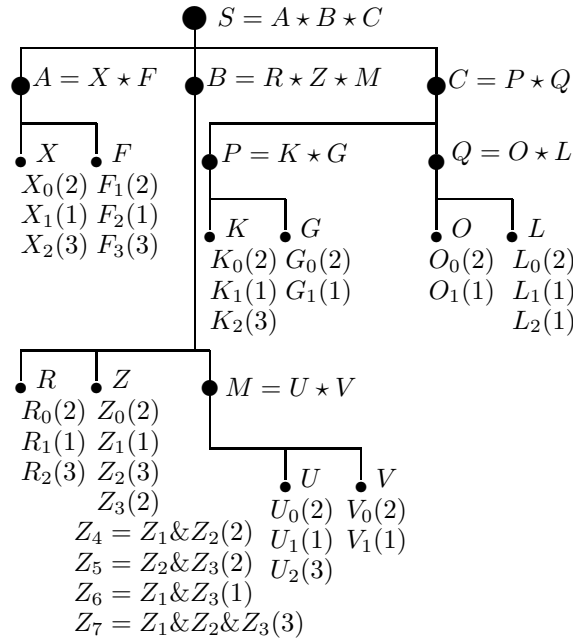


Fig. 27. Structure of repair plan [132]

Table 5. Compatibility [132]

	$M_1$	$M_2$	$R_0$	$R_1$	$R_2$
$Z_0$	2	3	3	3	0
$Z_1$	3	2	2	3	3
$Z_2$	3	2	0	3	3
$Z_3$	3	2	0	2	3
$Z_4$	3	2	2	3	3
$Z_5$	3	2	0	3	3
$Z_6$	3	2	2	3	3
$Z_7$	3	2	2	3	3
$M_1$			0	3	3
$M_2$			3	2	2

	$L_0$	$L_1$	$L_2$
$O_0$	3	2	2
$O_1$	1	3	3

	$F_1$	$F_2$	$F_3$
$X_1$	3	3	3
$X_2$	3	3	3
$X_3$	0	3	2

Table 6. Compatibility [132]

	$G_0$	$G_1$
$K_0$	3	0
$K_1$	2	3
$K_2$	1	2

	$V_0$	$V_1$
$U_0$	3	0
$U_1$	0	2
$U_2$	0	3

The evident design scheme for morphological hierarchy is the following:

*Stage 1.* Design of system hierarchy (one of the methods above).

*Stage 2.* Generation of DAs for each leaf node of the system model (expert judgement and/or usage of information bases).

*Stage 3.* Generation of scales for assessment of DAs and assessment of DAs (usually multicriteria description of DAs is used at a preliminary phase).

*Stage 4.* generation of an ordinal scale for compatibility among DAs and assessment of the compatibility.

### 3. SOME APPROACHES TO MODIFICATION

#### 3.1. Modification of Tree via Condensing of Weighted Edges

This section <sup>3</sup> describes transformation of a tree (with weights of vertices and weights of edge/arcs) via integration (condensing) of some neighbor vertices while taking into account a constraint for a total weight of the maximum tree tail (i.e., length from root to a leaf vertex). The problem was firstly formulated for designing an overlay structure of a modular software system in [121]. The integration of software modules requires additional memory, but allows to decrease a time (i.e., frequency) of loading some corresponding modules. Other applications of the problem can be examined as well, e.g., hierarchical structure of data, call problem, hierarchical information structure of Web-sites. This problem is illustrated in Fig. 28 and Fig. 29 by an example for designing the over-layer structure on the basis of module integration, when different software or data modules can apply the same parts of RAM. A new kind of FPTAS for the above-mentioned combinatorial optimization problem (a generalization of multiple choice problem over a tree-like structure and special constraints) was suggested in [121].

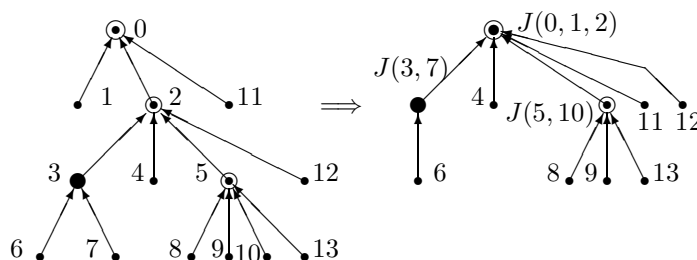


Fig. 28. Integration of software modules (over-layer structure)

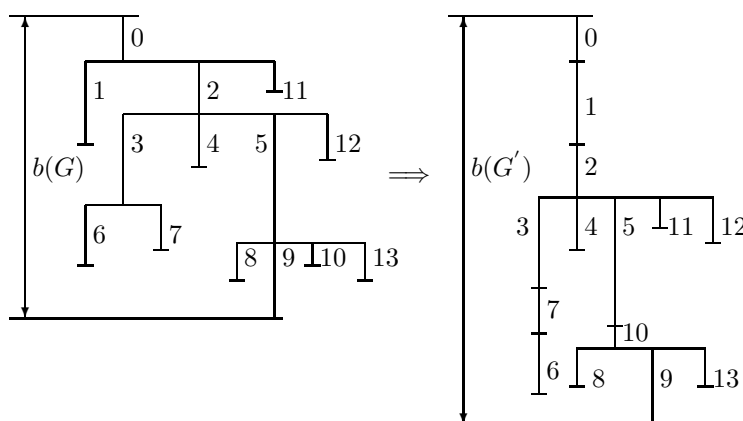


Fig. 29. Usage of memory (RAM)

Let  $G = (A, \Upsilon)$  be an oriented tree, where  $A$  is a set of vertices (software or data modules).  $\Upsilon$  is a multi-valued mapping of  $A$  into  $A$ . Arcs of  $G$  are oriented from the root  $a_o \in A$  to leaf vertices. Each vertex  $a \in A$  has a positive weight (required volume of RAM)  $\beta(a) > 0$ . Each arc  $(a', a'')$  ( $a', a'' \in A$ ,  $a'' \in \Upsilon a'$ ) has a weight (i.e., an initial frequency of loading into RAM)  $w(a', a'') > 0$ . This arc weight corresponds to the frequency of calling (and loading) from module  $a'$  to module  $a''$ .

<sup>3</sup> From (with amendments):

(i) M.Sh. Levin, An extremal problem of organization of data. *Eng. Cybern.*, 19(5), 1981, 87-95.

(ii) M.Sh. Levin, *Combinatorial Engineering of Decomposable Systems*, Springer, 1998, Chapter 2.

Let  $\pi(a^1, a^l) = \langle a^1, \dots, a^i, \dots, a^l \rangle$  be a path ( $a^{j+1} \in \Upsilon a^j, j = 1, \dots, l-1$ ). We propose for each path a weight  $\lambda(\pi(a^1, a^l)) = \sum_{i=1}^l \lambda(a^i)$ . Denote by a weight of graph  $G$  the value

$$\lambda(G) = \max_{a'' \in A^o} \{\lambda(\pi(a_o, a''))\},$$

where  $A^o = \{a \in A \mid \Upsilon a = \emptyset\}$  is a set of leaf vertices. Let  $G_a = (A_a, \Upsilon)$  is a subtree with root  $a \in A$ , and  $A_a$  contains vertex  $a$  and all other vertices, which can be reached from  $a$ . Graph  $(A_a \setminus a, \Upsilon)$  is called *tail* of vertex  $a$ , and value  $\lambda^-(a) = \lambda(G_a) - \lambda(a)$  is called a *tail* weight of vertex  $a$ . Clearly that

$$\lambda(a) = \max_{a' \in \Upsilon a} \{\lambda(G_{a'})\}.$$

We examine weight  $w(a)$  and binary variable  $x(a) \forall a \in A \setminus a_o$  (1 corresponds to a situation when the arc, directed to  $a$ , is condensed). Now let us define a transformation of graph  $G$  on the basis of integrating the vertices  $a'$  and  $a''$  as follows:

- (a) vertex  $a'$  is changed into  $J(a', a'')$  with the following properties:  
 $\lambda(J(a', a'')) = \lambda(a') + \lambda(a'') \quad \Upsilon J(a', a'') = (\Upsilon a' \cup \Upsilon a'') \setminus a''$ ;
- (b) vertex  $a''$  and arcs, which are oriented from the vertex, are deleted.

For graph  $G$ , we propose a binary vector  $\kappa(a)$  that involves all  $x(a) \forall a \in A \setminus a_o$ . Thus, we examine the weights of vertex  $a$  and its *tail* as functions of vector  $\kappa$ :  $\lambda(a, \kappa), \lambda^-(a, \kappa)$ . Now let us consider a problem (kind 1):

$$\max W(\kappa) = \sum_{a \in A \setminus a_o} x(a)w(a) \quad \text{s.t.} \quad \lambda(a_o, \kappa) + \lambda^-(a_o, \kappa) \leq b,$$

where  $b$  is a positive constant (i.e., a volume of accessible RAM). In general, this problem formulation corresponds to the example in Fig. 28 and Fig. 29.

In addition, we examine analogical problem (kind 2) with other constraints as follows:

$$\lambda(a_o, \kappa) \leq b^-, \quad \lambda^-(a_o, \kappa) \leq b^+, \quad b^- + b^+ = b.$$

Note, illustrations of the class of considered combinatorial problems are presented in Fig. 30 (basic knapsack problem and multiple choice problem) and in Fig. 31, correspondence of problem to illustration is pointed out in Table 7.

Now consider some simple cases of the problems (kind 1 and kind 2). Let  $\Upsilon a_o = \{a_1, \dots, a_i, \dots, a_m\}$  (and  $u_i$ ) corresponds to an arc  $(a_o, a_i)$  ( $w(u_i) = w_i$ ). Then, corresponding problem (problem 1, an equivalent to knapsack problem, Fig. 30i) is:

$$\max \sum_{i=1}^m x_i w_i \quad \text{s.t.} \quad \lambda(a_o) + \sum_{i=1}^m x_i \lambda(a_i) \leq b, \quad x_i \in \{0, 1\}.$$

The objective function in other simple cases (1.1 - Fig. 31a, 1.2 - Fig. 31b, 1.3 - Fig. 31c, 1.4 - Fig. 31d), which are based on knapsack problem (problem 1, Fig. 30i) is analogical, and only constraints will be presented for them.

Problem 1.1 (Fig. 31a) has the following constraint:

$$\lambda(a_o) + \sum_{i=1}^m x_i \lambda(a_i) + \max_{1 \leq i \leq m} ((1 - x_i) \lambda(a_i)) \leq b.$$

This problem corresponds to a “kernel” load in many software packages.

Problem 1.2 (Fig. 31b) of kind 2 is the following:

$$\lambda(a_o) + \sum_{i=1}^m x_i \lambda(a_i) \leq b^-, \quad \max_{1 \leq i \leq m} ((1 - x_i) \lambda(a_i)) \leq b^+.$$

Problem 1.3 (Fig. 31c) is:

$$\lambda(a_o) + \sum_{i=1}^m x_i \lambda^-(a_i) + \max_{1 \leq i \leq m} ((1 - x_i) \lambda^-(a_i)) + \lambda^+(a_i) \leq b.$$

It is reasonable to point the following properties of this problem:

- (a)  $a_i$  ( $\forall a_i \in \Upsilon a_o$ ) has the weight  $\lambda^-(a_i)$ ;
- (b)  $a_i$  ( $\forall a_i \in \Upsilon a_o$ ) has the only one son with a weight  $\lambda^+(a_i)$ , and the value is the *tail* weight; and
- (c) only condensing the following arcs  $(a_o, a_i)$  ( $i = \overline{1, m}$ ) is admissible.

As a result, a sequence of simple problems based on knapsack problem can be examined: 1 (basic knapsack problem, Fig. 30i), 1.1 (analogue of knapsack problem, Fig. 31a), 1.2 (Fig. 31b), 1.3 (Fig. 31c), 1.4 (Fig. 31d).

In the same way, a sequence of auxiliary problems based on multiple choice problem can be considered: 2 (basic multiple choice problem, Fig. 30ii), 2.1 (analogue of multiple choice problem, Fig. 31e), 2.2 (Fig. 31f), 2.3 (Fig. 31g), 2.4 (Fig. 31h).

In our case, multiple choice problem or problem 2 (Fig. 28ii) is the following:

$$\max W(\{x_{ij}\}) = \sum_{i=1}^m \sum_{j=1}^{q_i} w(a_{ij}) x_{ij} \quad s.t. \quad \lambda(a_o) \sum_{i=1}^m \sum_{j=1}^{q_i} x_{ij} \lambda(a_{ij}) \leq b, \quad \sum_{j=1}^{q_i} x_{ij} = 1, \quad i = \overline{1, m}; \quad x_{ij} \in \{0, 1\}.$$

Here, the following set of Boolean vectors in auxiliary problems is used:

$$X = \{\kappa = (x_{ij}^1; x_{ij}^2) | x_{ij}^1, x_{ij}^2 \in \{0, 1\}; \quad j = \overline{1, q_i}; \quad i = \overline{1, m}\}$$

In addition, the following constraint has to be taken into account in all auxiliary problems:

$$\sum_{j=1}^{q_i} x_{ij}^1 = 1, \quad \forall i; \quad x_{ij}^2 \leq x_{ij}^1, \quad \forall i, j.$$

Also, the following modified objective function is used:

$$W(X) = \sum_{i=1}^m \sum_{j=1}^{q_i} (x_{ij}^1 w^-(a_{ij}) + x_{ij}^2 w(a_{ij})).$$

Now, for example, auxiliary problem 2.4 is considered that corresponds to kind 2 above (Fig. 31h):

$$\lambda(a_o) + \sum_{i=1}^m \sum_{j=1}^{q_i} x_{ij}^2 \lambda^-(a_{ij}) \leq b^-, \quad \max_{i,j} ((1 - x_{ij}^2) \lambda^-(a_{ij}) + \lambda^+(a_{ij})) \leq b^+.$$

For the sequence of simple problems above, we can apply approximation algorithms, which are based on an  $\epsilon$ -approximate algorithm ( $\epsilon \in [0, 1]$ ) for knapsack problem (e.g., [110], [156], [183], [184]). In the case of these algorithms, an estimate of an operation number is similar for knapsack problem (e.g., [110], [156]), and equals  $O(\frac{m^2}{\epsilon})$  [121]. The algorithms apply ordering of elements from set  $\Upsilon a$  by non-decreasing of  $\lambda(a_i)$  or  $(\lambda^-(a_i) + \lambda^+(a_i))$ .

The solving process of auxiliary problems is based on similar approximation approach to multiple choice problem with the following estimates of number of operations and required memory accordingly (e.g., [110], [156]):  $O(\frac{m}{\epsilon} \sum_{i=1}^m q_i)$ ,  $O(\frac{m^2}{\epsilon} \max_{1 \leq i \leq m} \{q_i\})$ . Unfortunately, we could not construct an algorithm with similar estimates for the auxiliary problem 2.3 (Fig. 31g) [121]. As a result,  $(\epsilon, \delta)$ -approximate algorithms with the following estimates (number of operations, and required memory) were suggested [121]:  $O(\frac{m}{\epsilon \delta} \sum_{i=1}^m q_i)$ ,  $O(\frac{m^2}{\epsilon} \max_{1 \leq i \leq m} \{q_i\})$ , where  $\delta$  is a relative error for constraints.

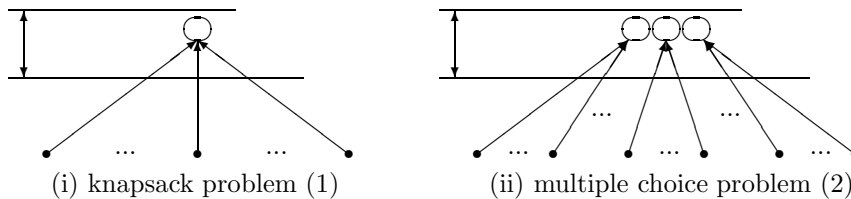


Fig. 30. Illustration for knapsack and multiple choice problems [123]

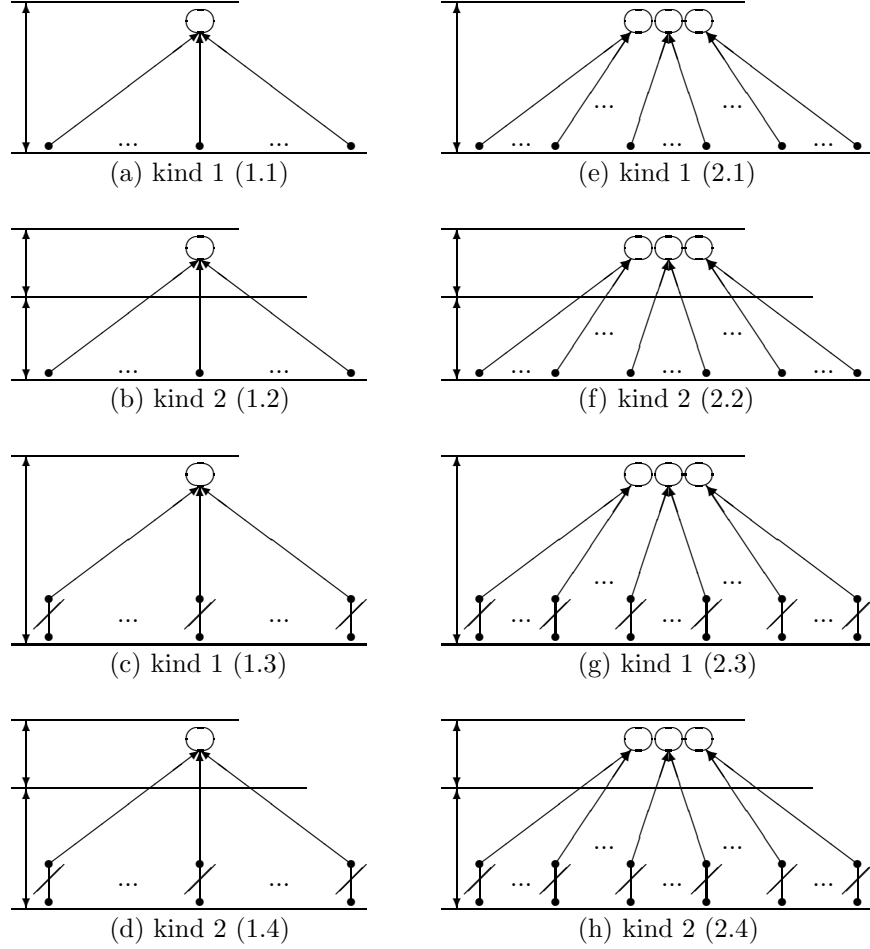


Fig. 31. Illustration for simplest and auxiliary problems [123]

Table 7. Knapsack-like problems - illustrations

Problem	Illustration
4.Basic knapsack problem ( <i>problem 1</i> )	Fig. 30i
5. <i>Problem 1.1</i>	Fig. 31.a
5. <i>Problem 1.2</i>	Fig. 31.b
5. <i>Problem 1.3</i>	Fig. 31.c
5. <i>Problem 1.4</i>	Fig. 31.d
4.Multiple choice problem ( <i>problem 2</i> )	Fig. 30ii
5. <i>Problem 2.1</i>	Fig. 31.e
5. <i>Problem 2.2</i>	Fig. 31.f
5. <i>Problem 2.3</i>	Fig. 31.g
5. <i>Problem 2.4</i>	Fig. 31.h

When  $G$  is a  $k$ -level tree, the algorithm is based on cascade-like 'Bottom-Up' process (Fig. 32) [121]:

*Step 1.* Problem 1.2.

*Step  $j$*  ( $j = 2, k-2$ ). Problem 2.4

*Step  $(k-1)$ .* Problem 2.3.

Estimates of the algorithms are as follows (i.e., operations, and memory):  $O(\frac{n^2\eta^5(a_o)}{\epsilon\delta^4})$ ,  $O(\frac{m^2\eta^4}{\epsilon\delta^4})$ , where  $m(a) = |\Upsilon(a)|$ ,  $m = \max_{a \in A} m(a)$ ,  $\eta(a) = |A_a \setminus \{a' \in A_a \mid \Upsilon a' = \emptyset\}|$ .

In the case of 3-level tree, the estimate of the operation number is:  $O(\frac{n^2\eta^4(a_o)}{\epsilon\delta^3})$ .

Prospective generalizations of the problem may involve the following: (a) multicriteria descriptions of the elements, (b) more complicated structure (e.g., parallel-series graph), (c) uncertainty, and (d) dynamics.

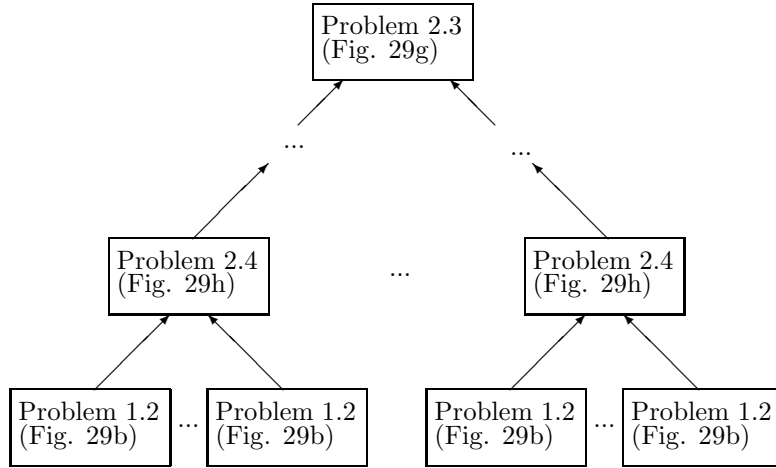


Fig. 32. 'Bottom-Up' solving scheme for tree-like case [123]

### 3.2. Hotlink Assignment Problem

In recent decade, the concept of “hotlinks” has been introduced for decreasing the complexity of access in web directories (or similar information structures) via usage (inserting) of a limited set of additional hyperlinks (i.e., “hotlinks”) to data (e.g., [26],[53],[171]). In general, “hotlink assignment problem” is a network upgrade problem (e.g., [69]):

*Find additional new arc(s) to the initial graph in order to insert shortcuts and decrease the expected path length.*

Mainly, the problem is examined for trees. Let  $T = (A, E)$  be a directed tree with maximum degree  $d$ , rooted at a node  $r_0 \in A$  (elements of  $A$  correspond to Web sites, elements of  $E$  correspond to hyperlinks). A node weight equals its access (search) frequency (probability). It is assumed that required information is contained at leaf nodes (for simplicity). The length of the search for node  $v \in A$  equals the number of links in the path from  $r_0$  to  $v$ .

Let  $T_u = (A_u, E_u)$  be a subtree of  $T$  ( $A_u \subseteq A, E_u \subseteq E$ ), rooted at node  $u \in A$  (here  $u$  is not the son of  $r_0$ ). Thus, additional direct link (“hotlink”) will be as follows:  $(r_0, u)$ . In this case, a path to all leaf nodes in  $T_u$  will be smaller.

Fig. 33, Fig. 34, Fig. 35, and Fig. 36 illustrate the simplest versions of “hotlink assignment” problem. Note, Fig. 36 depicts the usage of internal nodes as additional roots (i.e.,  $r_1$ ).

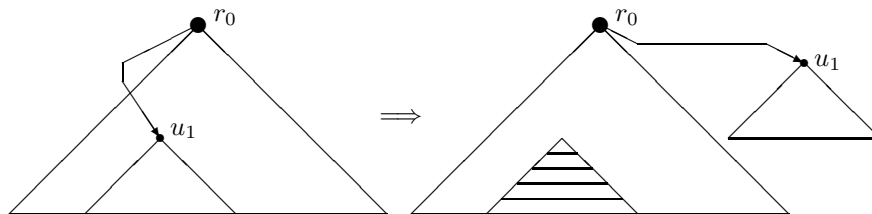


Fig 33. Hotlink assignment problem (one hotlink)

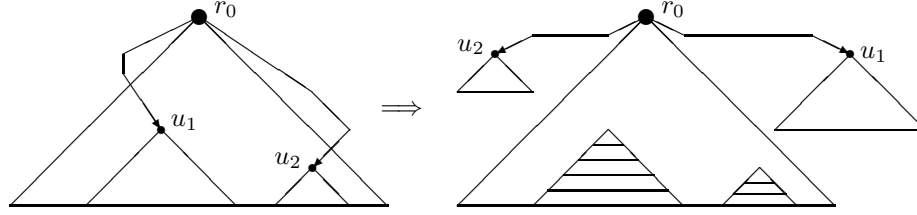


Fig 34. Hotlink assignment problem (two hotlinks)

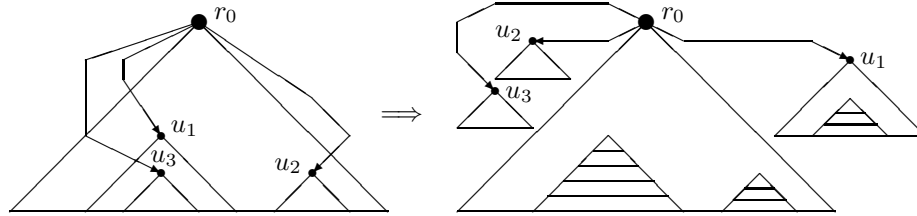


Fig 35. Hotlink assignment problem (three hotlinks)

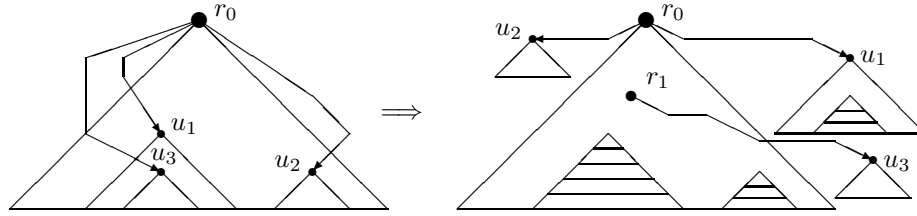


Fig 36. Hotlink assignment problem (three hotlinks, one internal root)

The basic “hotlink assignment” problem consists in assignment of  $k$  additional “hotlinks” (from the root) to minimize the total number of steps to visit the required information nodes. On the other hand, it is necessary to find the set of  $k$  nodes ( $U = \{u\}$ ) at the tree  $T$ . In recent years, various “hotlink assignment” problems have been intensively studied (e.g., algorithm design, issues of complexity, approximation) (e.g., [26],[46],[52],[53], [69],[104],[117], [119],[157],[171]). Some versions of the problem are presented in Table 8. Mainly, “hotlink assignment” problems belong to class of NP-hard problems (e.g., [104]). Many approximation algorithms have been suggested for the problems (including FPTAS) (e.g., [116], [117], [157]).

In the case of multiple attribute description of “hotlinks” or/and selected nodes/subtrees (i.e., nodes as  $u$ ), multicriteria knapsack like problems or multicriteria generalized assignment problems may be used (e.g., [138], [139], [150], [170]). It may be reasonable to examine some generalizations of the “hotlink assignment” problem, for example: (i) taking into account uncertainty, (ii) tree-inclusion problem (as “hot-tree assignment” problem). Application of “hotlink assignment” problems is very useful for many domains, e.g., adaptive Web sites systems, knowledge bases, file systems, menus systems, asymmetric communication protocols (e.g., [26], [46], [104], [119], [171]).

Table 8. Hotlink assignment problems

Problem	Source
1.Basic hotlink assignment	[26],[171]
2.Single hotlink assignment	[53],[116]
3.Hotlinks only for leafs	[104]
4.Multiple hotlink assignment	[53],[69]
5.Dynamic hotlink assignment	[52]

Generally, “hotlink assignment” problem is a special case of “graph augmentation problem” (e.g.,



[63],[111]). Here the goal is to modify an initial graph (e.g., by edges) such that the augmented graph will be satisfied some requirements (e.g., as increasing the connectivity).

### 3.3. Scheme for Transformation of Tree to Steiner Tree

Here a transformation of a tree  $T = (A, E)$  into Steiner tree  $S = (A', E')$  is considered as addition of Steiner points into an initial tree (or a preliminary built spanning tree) while taking into account the following: “cost” (required resource) of each Steiner point, “profit” of each Steiner point, total resource constraint (i.e., total “cost” of the selected Steiner points) [141]. A simplest case is considered when Steiner points for triangles are only examined. Evidently, vector-like “cost” and “profit” can be used as well. The solving scheme is the following:

*Stage 1.* Identification (e.g., expert judgment, clustering) of  $m$  regions (clusters, groups of neighbor nodes) in the initial tree  $T$  for possible addition of Steiner points.

*Stage 2.* Generation of possible Steiner points (candidates) and their attributes (i.e., cost of addition, “profit”).

*Stage 3.* Formulation of multiple choice problem for selection of the best additional Steiner points while taking into account resource constraint(s):

$$\max \sum_{i=1}^m \sum_{j=1}^{q_i} c_{ij} x_{ij} \quad s.t. \quad \sum_{i=1}^m \sum_{j=1}^{q_i} a_{ij} x_{ij} \leq b, \quad \sum_{j=1}^{q_i} x_{ij} = 1, \quad x_{ij} \in \{0, 1\};$$

where  $i$  is the index of region ( $i = \overline{1, m}$ ),  $q_i$  is the number of versions for addition of Steiner points in region  $i = \overline{1, m}$ ,  $j$  is the index of version for addition of Steiner points in region ( $j = \overline{1, q_i}$ ),  $x_{ij}$  is binary variable that equals 1 if version  $j$  in region  $i$  is selected,  $b$  is a total constraint for the required resources (i.e., a total “cost”).

*Stage 4.* Solving the multiple choice problem to obtain the resultant Steiner tree  $S$ .

A numerical illustrative example illustrates the scheme. Initial tree is (Fig. 37):  $T = (A, E)$ ,  $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$ . Four regions are defined (Fig. 37): region 1:  $\{1, 2, 3, 4\}$ ; region 2:  $\{4, 6, 7\}$ ; region 3:  $\{4, 5, 6, 9, 11\}$ ; and region 4:  $\{7, 8, 10\}$ . The considered Steiner points are the following (Fig. 38): region 1:  $s_{11}, s_{12}$ ; region 2:  $s_{21}$ ; region 3:  $s_{31}, s_{32}$ ; and region 4:  $s_{41}$ . Table 9 contains for multiple choice problem: binary variables and corresponding attributes (required resource as “cost”, possible “profit”). Thus, the problem is:

$$\max \sum_{i=1}^4 \sum_{j=1}^{q_i} c_{ij} x_{ij} \quad s.t. \quad \sum_{i=1}^4 \sum_{j=1}^{q_i} a_{ij} x_{ij} \leq b, \quad \sum_{j=1}^{q_i} x_{ij} = 1, \quad x_{ij} \in \{0, 1\}.$$

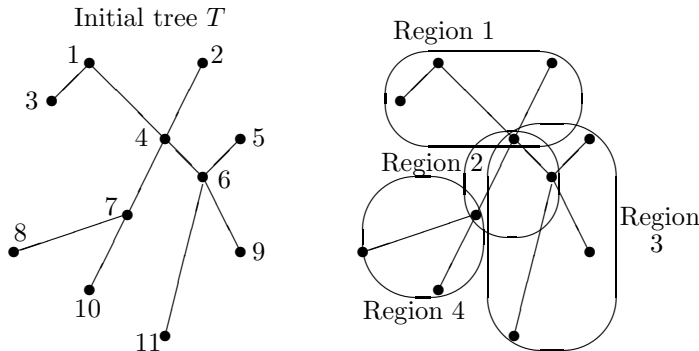


Fig. 37. Initial tree and regions (clusters)

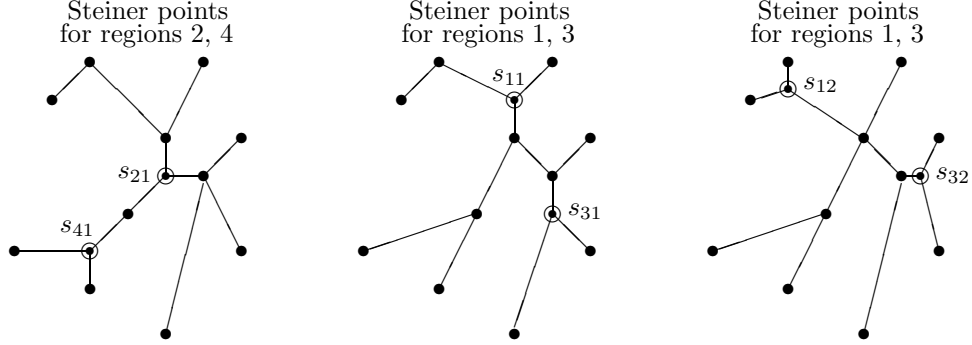


Fig. 38. Additional Steiner points for regions

Table 9. Data for multiple choice problem

Region	Binary variable	Steiner point	"Cost" $c_{ij}$	"Profit" $a_{ij}$
Region 1	$x_{11}$	None	0.0	0.0
	$x_{12}$	$s_{11}$	3.1	1.5
	$x_{13}$	$s_{12}$	1.2	1.4
Region 2	$x_{21}$	None	0.0	0.0
	$x_{22}$	$s_{21}$	2.0	1.3
Region 3	$x_{31}$	None	0.0	0.0
	$x_{32}$	$s_{31}$	2.4	1.4
	$x_{33}$	$s_{32}$	1.8	1.3
Region 4	$x_{41}$	None	0.0	0.0
	$x_{42}$	$s_{41}$	1.5	1.2

Some obtained solutions (i.e., as set of additional Steiner points) are the following (a simple greedy heuristic was used) (Fig. 39):

- (1)  $b_1 = 2.9$ :  $\bar{x}_{b_1}$ :  $x_{12} = 1$ ,  $x_{21} = 1$ ,  $x_{32} = 1$ ,  $x_{41} = 1$ , Steiner points  $Z_{b_1} = \{s_{11}, s_{31}\}$ , total (additive) "profit"  $\bar{c} = 5.5$ ;
- (2)  $b_2 = 4.2$ :  $\bar{x}_{b_2}$ :  $x_{12} = 1$ ,  $x_{22} = 1$ ,  $x_{32} = 1$ ,  $x_{42} = 0$ ; Steiner points  $Z_{b_2} = \{s_{11}, s_{21}, s_{31}\}$ , total (additive) "profit"  $\bar{c} = 7.5$ ;
- (3)  $b_3 = 5.4$ :  $\bar{x}_{b_3}$ :  $x_{12} = 1$ ,  $x_{22} = 1$ ,  $x_{32} = 1$ ,  $x_{42} = 1$ , Steiner points  $Z_{b_3} = \{s_{11}, s_{21}, s_{31}, s_{41}\}$ , total (additive) "profit"  $\bar{c} = 9.0$ .

Note, more complicated problem can be examined, for example, while taking into account the following:

(i) several additional Steiner points in the same region, (ii) compatibility of points additions in neighbor regions.

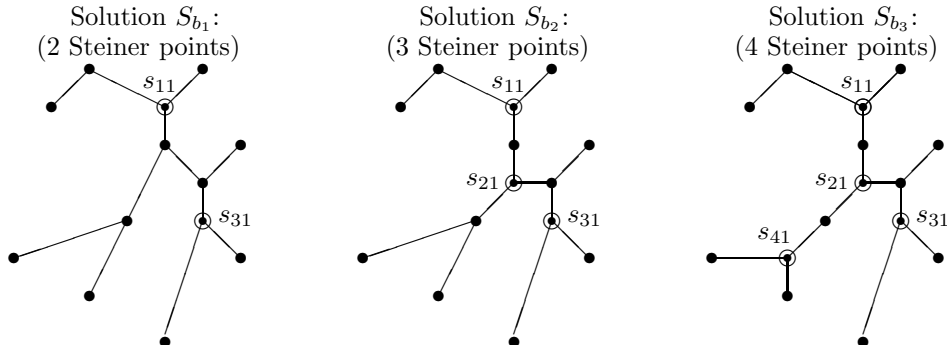


Fig. 39. Solutions

### 3.4. Towards Restructuring Problems

In recent several years, a special class of combinatorial optimization problems as “reoptimization” has been intensively studied for several well-known problems: (a) minimum spanning tree problem [25], (b) traveling salesman problems ([8], [10]), (c) Steiner tree problems ([19],[67]), (d) covering problems [20].

In general, the reoptimization problem is formulated as follows:

*Given:* (i) an instance of the combinatorial problem over a graph and corresponding optimal solution, (ii) some “small” perturbations (i.e., modifications) on this instance (e.g., node-insertion, node-deletion).

*Question:* Is it possible to compute a new good (optimal or near-optimal) solution subject to minor modifications?

A survey of complexity issues for reoptimization problems is presented in [24]. Mainly, the problems belong to class of NP-hard problems and various approximation algorithms have been suggested

In [129], the author has suggested another approach to modification in combinatorial optimization problems as “restructuring”. The approach corresponds to many applied reengineering (redesign) problems in existing modular systems. The restructuring process is illustrated in Fig. 40 [129]. Here modifications are based on insertion/deletion of elements (i.e., elements, nodes, arcs) and changes of a structure as well. Two main features of the restructuring process are examined: (i) a cost of the initial problem solution restructuring (i.e., cost of the selected modifications), (ii) a closeness the obtained restructured solution to a goal solution.

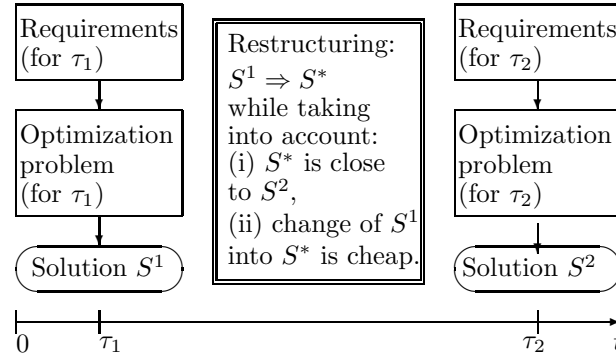


Fig. 40. Illustration for restructuring process [129]

This kind of problems corresponds to redesign/reconfiguration (improvement, upgrade) of modular systems and the situations can be faced in complex software, algorithm systems, communication networks, computer networks, information systems, manufacturing systems, constructions, etc. (e.g. [127], [129]). The optimization problem is solved for two time moments:  $\tau_1$  and  $\tau_2$  to obtain corresponding solutions  $S^1$  and  $S^2$ . The examined restructuring problem consists in a “cheap” transformation (change) of solution  $S^1$  to a solution  $S^*$  that is very close to  $S^2$ . In [129], this restructuring approach is described and illustrated for the following combinatorial optimization problems: knapsack problem, multiple choice problem, assignment problem, spanning tree problems. Evidently, this restructuring problem may be used for many combinatorial optimization problems as changing a solution (e.g., subset, structure). Fig. 41 depicts the restructuring problem [129].

Let  $P$  be a combinatorial optimization problem with a solution as structure  $S$  (i.e., subset, graph),  $\Omega$  be initial data (elements, element parameters, etc.),  $f(P)$  be objective function(s). Thus,  $S(\Omega)$  be a solution for initial data  $\Omega$ ,  $f(S(\Omega))$  be the corresponding objective function. Let  $\Omega^1$  be initial data at an initial stage,  $f(S(\Omega^1))$  be the corresponding objective function.  $\Omega^2$  be initial data at next stage,  $f(S(\Omega^2))$  be the corresponding objective function.

As a result, the following solutions can be considered: (a)  $S^1 = S(\Omega^1)$  with  $f(S(\Omega^1))$  and (b)  $S^2 = S(\Omega^2)$  with  $f(S(\Omega^2))$ . In addition it is reasonable to examine a cost of changing a solution into another one:  $H(S^\alpha \rightarrow S^\beta)$ . Let  $\rho(S^\alpha, S^\beta)$  be a proximity between solutions  $S^\alpha$  and  $S^\beta$ , for example,  $\rho(S^\alpha, S^\beta) = |f(S^\alpha) - f(S^\beta)|$ . Note, function  $f(S)$  is often a vector function. Finally, the restructuring problem can be examine as follows (a basic version):

Find a solution  $S^*$  while taking into account the following:

- (i)  $H(S^1 \rightarrow S^*) \rightarrow \min$ , (ii)  $\rho(S^*, S^2) \rightarrow \min$  (or constraint).

Thus, the basic optimization model can be examined as the following:

$$\min \rho(S^*, S^2) \quad s.t. \quad H(S^1 \rightarrow S^*) \leq \hat{h},$$

where  $\hat{h}$  is a constraint for cost of the solution change.

Proximity function  $\rho(S^*, S^2)$  can be considered as a vector function (analogically for the solution change cost). The situation will lead to a multicriteria restructuring problem (i.e., searching for a Pareto-efficient solutions).

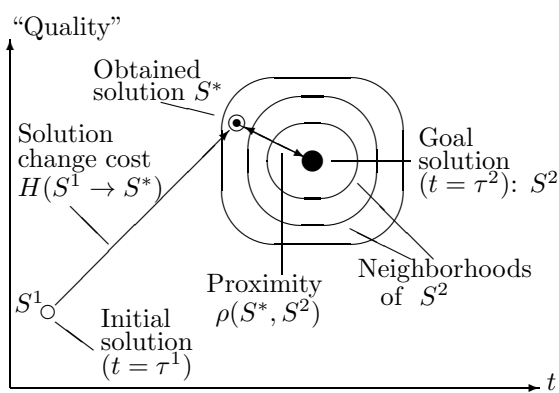


Fig. 41. Illustration for restructuring problem [129]

#### 4. CONCLUSION

The paper contains a generalized integrated glance to design of system hierarchies. The problems of this kind are often crucial ones in system analysis, in systems design. The following basic hierarchical structures are considered: trees, a special morphological hierarchy, multi-layer structures. Evidently, the examination is based on combinatorial optimization models. The list of design methods involves the following: expert-based 'top-down' procedure ('divisive' strategy of hierarchical clustering), hierarchical clustering (agglomerative algorithm), spanning trees, ontological approach, 'optimal' organization, multi-layer structures (including multilevel network design problem).

A special attention is targeted to modification problems. Here, a simplified modification of trees is examined (e.g., augmentation problem for graphs is not considered [63]). The presented modification problems are as follows: modification of a weighted tree via condensing neighbor nodes (i.e., design of over-lay structure of tree-like software), hotlink assignment problem, building of a Steiner tree based on an initial tree, new class of restructuring problems.

Generally, our material is the first integrated attempt to the problem. As a result, many considered issues require additional studies and analysis of many applied examples, many other significant problems have to be described and studied. For example, issues of uncertainty have not been examined. Mainly, considered problems are described on the basis of a simplified frames (e.g., engineering description, problem formulation, basic solving schemes, prospective versions).

The presented material is oriented to needs of applications. In the future, it may be interesting to use the described approaches in education (engineering, management, computer science and information technology, applied mathematics). This material can be considered and used as a tutorial.

#### REFERENCES

1. D. Abts, B. Felderman, A guided tour of data-center networking. *Commun. of the ACM* 55(6) (2012) 44–51.
2. A. Agrawal, P. Klein, R. Ravi, When trees collide: an approximation algorithm for the generalized Steiner problem on networks. *SIAM J. on Computing* 24(3) (1995) 440–456.

3. R. Agrawal, R. Srikant, On integrating catalogs. In: *Proc. 10th Int. World Wide Web Conf. WWW 2001*, Hong Kong, (2001) 603–612.
4. M.J. Aitkenhead, A co-evolution decision tree classification method. *Expert Systems with Applications* 34(1) (2008) 18–25.
5. N. Alon, B. Awerbuch, Y. Azar, N. Buchbinder, J. Naor, A general approach to online network optimization problems. *ACM Trans. on Algorithms* 2(4) (2006) 640–660.
6. N. Alon, F. Fomin, G. Gutin, M. Krivelevich, S. Saurabh, Spanning directed trees with many leaves. *SIAM J. on Discr. Math.* 23(1) (2009) 466–476.
7. A. Amir, D. Keselman, Maximum agreement subtree of a set of evolutionary trees - metrics and efficient algorithms. *SIAM J. on Comp.* 26(6) (1997) 1656–1669.
8. C. Archetti, L. Bertazzi, M.G. Speranza, Reoptimizing the traveling salesman problem. *Networks* 42(3) (2003) 154–159.
9. J.E.C. Arroyo, P.S. Vieira, D.S. Vianna, A GRAPS algorithm for the multi-criteria minimum spanning tree problem. *Ann. of Oper. Res.* 159(1) (2008) 125–133.
10. G. Austello, B. Escoffer, J. Monnot, V. Paschos, Reoptimization of minimum and maximum traveling salesman's tours. *J. of Discr. Algorithms* 7(4) (2009) 453–463.
11. B. Awerbuch, Y. Azar, Y. Bartal, On-line generalized Steiner problem. *Theoretical Comput. Sci.* 324(2-3) (2004) 313–324.
12. S.W. Bae, C. Lee, S. Choi, On exact solutions to the Euclidean bottleneck Steiner tree problem. *Information Processing Letters* 110(16) (2010) 672–678.
13. A. Balakrishnan, T. magnanti, P. Mirchandani, A dual-based algorithm for multi-level network design. *Manag. Sci.* 40(5) (1994) 567–581.
14. H.H. Baligh, *Organization Structures: Theory and Design, Analysis and Prescription*. Springer, 2006.
15. C. Batini, M. Lenzerini, S.B. Navathe, A comparative study of methodologies for data base schema integration. *ACM Computing Survey* 18(4) (1986) 323–364.
16. Belotserkovskii D.L., Vishnevskii V.M., New algorithm for generating a biconnected spanning subgraphs for topological optimization of data networks, *Autom. & Remote Control* 58(1) (part 2) (1997) 88–97.
17. C. Berge, *Graphs and Hypergraphs*. Elsevier, New York, 1973.
18. C. Berge, *Hypergraphs: Combinatorics of Finite Sets*. North Holland, Amsterdam, 1989.
19. D. Bilo, H.-J. Bockenhauer, J. Hromkovic, R. Kralovic, T. Momke, P. Widmayer, A. Zych, Reoptimization of Steiner trees. In: J. Gudmundsson (Ed.), *Proc. of Scandinavian Workshop on Algorithm Theory SWAT'08*, LNCS 5124, Springer, (2008) 258–269.
20. D. Bilo, P. Widmayer, A. Zych, Reoptimization of weighted graph and covering problems. In: E. Bampis, M. Skutella (Eds.), *Proc. of Workshop on Approximation and Online Algorithms WAOA'08*, LNCS 5426, Springer, (2008) 201–213.
21. A. Bley, T. Koch, R. Wessaly, Large-scale hierarchical networks: How to compute an optimal architecture, *Proc. of Networks 2004*, Vienna, 2004.
22. J. Blum, M. Ding, A. Thaeler, X. Cheng, Connected dominating set in sensor networks and MANETs. In: D.-Z. Du, P. Pardalos (Eds.), *Handbook of Combinatorial Optimization*, Kluwer, (2005) 329–369.
23. K. Bogdanov, M. Holcombe, Statechart testing method for aircraft control systems. *Software Testing, Verification and Reliability* 11(1) (2001) 39–54.
24. H.-J. Bockenhauer, J. Hromkovic, T. Momke, P. Widmayer, on the hardness of reoptimization. In: *Proc. of SOFSEM 2008 - Theory and Practice of Informatics*, LNCS 4910, Springer, (2008) 50–65.
25. N. Boria, V.Ph. Paschos, Fast reoptimization for the minimum spanning tree problem. *J. of Discr. Algorithms* 8(3) (2010) 296–310.
26. P. Bose, J. Czyzowicz, L. Gasieniec, E. Kranakis, D. Krizanc, A. Pelc, M.V. Martin, Strategies for hotlink assignments. *Int. Symp. on Algorithms and Computation (ISAAC'00)*, LNCS 1969, Springer, (2001) 23–34.
27. R.A. Botafofo, E. Rivlin, B. Shneiderman, Structural analysis of hypertexts: Identifying hierarchies and useful metrics. *ACM Trans. on Information Systems* 10(2) (1992) 83–110.
28. S. Butenko, X. Cheng, C. Oliveira, P.M. Pardalos, A new heuristic for the minimum connected dominating set problem on ad hoc wireless networks. In: *Recent Developments in Cooperative Control and Optimization*. Kluwer, (2004) 61–73.

29. S.A. Canuto, M.G.C. Resende, C.C. Ribeiro, Local search with perturbations for the prize-collecting Steiner tree problem in graphs. *Networks* 38(1) (2001) 50–58.
30. Y. Caro, D.B. West, R. Yuster, Connected domination and spanning trees with many leaves. *SIAM J. on Discr. Math.* 13(2) (2000) 202–211.
31. C. Chekuri, G. Even, G. Kortsarz, A greedy approximation algorithm for the group Steiner problem. *Discr. Appl. Math.* 154(1) (2006) 15–34.
32. D. Chen, D.-Z. Du, X.-D. Hu, G.-H. Lin, L. Wang, G. Xue, Approximations for Steiner trees with minimum number of Steiner points. *J. of Global Optimization* 18(1) (2000) 17–33.
33. G. Chen, S. Chen, W. Guo, W. Chen, The multicriteria minimum spanning tree problem based genetic algorithm, *Inform. Sci.* 177(22) (2007) 5050–5063.
34. H. Chen, K.J. Lynch, K. Basu, N.T. Hg, Integrating, and activating thesauri for concept-based document retrieval. *IEEE Expert* 8(2) (1993) 25–34.
35. H. Chen, B. Schatz, T. Hg, J. Martinez, A. Kirchhoff, C. Lin, A parallel computing approach to creating engineering concept spaces for semantic retrieval: The Illinois Digital Library Initiative Project. *IEEE Trans. PAMI* 18(8) (1996) 771–782.
36. X. Cheng, X. Huang, D. Li, W. Wu, D.-Z. Du, A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks* 42(4) (2003) 202–208.
37. S. Chorpá, E.R. Gorres, M.R. Rao, Solving the Steiner tree problem on a graph using branch and cut. *ORSA J. Comput.* 4(3) (1992) 320–355.
38. S. Chorpá, C. Tsai, A branch-and-cut approach for minimum cost multi-level network design. *Discr. Math.* 242(1-3) (2002) 65–92.
39. R.C. Conant, Information flows in hierarchical systems. *Int. J. of General Systems* 1(1) (1974) 9–18.
40. W.D. Cook, L.M. Seiford, M.Kress, A general framework for distance-based consensus in ordinal ranking models. *EJOR* 96(2) (1996) 392–397.
41. O. Corcho, M. Fernandez-Lopez, A. Gomez-Perez, Methodologies, tools and languages for building ontologies: Where is their meeting point? *Data & Knowledge Engineering* 46(1) (2003) 41–64.
42. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, *Introduction to Algorithms*, 2nd ed., Boston: MIT Press and McGraw-Hill, 2001.
43. A.M. Costa, J.-F. Cordeau, G. Laporte, Steiner tree problems with profits. *INFOR* 44(2) (2006) 99–115.
44. A.M. Costa, J.-F. Cordeau, G. Laporte, Fast heuristics for the Steiner tree problem with revenues, budget and hop constraints. *EJOR* 190(1) (2008) 68–78.
45. J. Current, C. ReVelle, J. Cohon, The hierarchical network design problem. *EJOR* 27(1) (1986) 57–66.
46. J. Czyzowicz, E. Kranakis, D. Krizanc, A. Pelc, M. Vargas Martin, Evaluation of hotlink structure for improving web performance. *J. Web Eng.* 1(2) (2003) 93–127.
47. F. Dai, J. Wu, An expected localized algorithm for connected dominating set formation in ad hoc wireless networks. *IEEE Trans. Parallel. Distrib. Syst.* 15(10) (2004) 908–920.
48. A. Dan, D.M. Dias, R. Kearney, T.C. Lau, T.N. Nguyen, F.N. Parr, M.W. Sachs, H.H. Shaikh, Business-to-business integration with tpaML and a business-to-business protocol framework. *IBM Systems Journal* 40(1) (2001) 68–90.
49. M.P. de Aragao, E. Uchoa, R.F. Werneck, Dual heuristics on the exact solution of large Steiner problems. *Electronic Notes in Discr. Math.* 7 (2001) 150–153.
50. J.W. Dean, Se Joon Yoon, G.I. Susman, Advanced manufacturing technology and organization structure: Empowerment or subordination? *Organization Science* 3(2) (1992) 203–229.
51. M. Dell’Amico, F. Maffioli, Combining linear and non-linear objectives in spanning tree problems. *J. of Combinatorial Optimization* 4(2) (2000) 253–269.
52. K. Douieb, S. Langerman, Dynamic hotlinks. *Algorithmica* 50(2) (2008) 208–222.
53. K. Douieb, S. Langerman, Near-entropy hotlink assignments. *Algorithmica* 58(2) (2010) 221–344.
54. D.E. Drake, S. Hougardy, On approximation algorithms for the terminal Steiner tree problem. *Information Processing Letters* 89(1) (2004) 15–18.
55. M. Drescher, A. Vetta, An approximation algorithm for the maximum leaf spanning arborescence problem. *ACM Trans. on Algorithms* 6(3) (2000) Art. 46.
56. M. Dror, M. Haouari, Generalized Steiner problems and other variants. *J. of Combinatorial Optimization* 4(4) (2000) 415–436.

57. M. Dror, M. Haouari, J. Chaouachi, Generalized spanning trees. *EJOR* 120(3) (2000) 583–592.
58. D.-Z. Du, L. Wang, B. Xu, The Euclidean bottleneck Steiner tree and Steiner tree with minimum number of Steiner points. In: *LNCS 2108*, Springer, (2001) 509–518.
59. D.-Z. Du, X. Hu, *Steiner Tree Problems in Computer Communication Networks*, Singapore: World Sci., 2008.
60. C.W. Duin, A. Volgenant, S. Voss, Solving group Steiner problems as Steiner problem. *EJOR* 154(1) (2004) 323–329.
61. R. Duncan, What is the right organizational structure? Decision tree analysis provides the answer. *Organizational Dynamics* 7(3) (1979) 59–80.
62. J.L. England, Hierarchies, theories, and methodologies. In: R. Trappl (Ed.) *Cybernetics and Systems'88*, Kluwer, (1988) 271–278.
63. K.P. Eswaran, R.E. Tarjan, Augmentation problems. *SIAM J. on Computing* 5(4) (1976) 653–665.
64. M. Feldman, G. Kortsarz, Z. Nutov, Improved approximating algorithms for the Directed Steiner Forest. In: *Proc. of the 20th Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, SIAM, Philadelphia, (2009) 922–931.
65. A. Fernandez, S. Gomez, Solving non-uniqueness in agglomerative hierarchical clustering using multidendrograms. *J. of Classification* 25(1) (2008) 43–65.
66. H. Fernau, J. Kneis, D. Kratsch, A. Langer, M. Liedloff, D. Raible, P. Rossmanith, An exact algorithm for the Maximum Leaf Spanning Tree problem. *Theor. Computer Science* 412(45) (2011) 6290–6302.
67. B. Escoffier, M. Milanic, V.Th. Paschos, Simple and fast reoptimizations for Steiner tree problem. *Algorithmic Operations Research* 4(2) (2009) 86–94.
68. B. Fuchs, A note on the terminal Steiner tree problem. *Information Processing Letters* 87(4) (2003) 219–220.
69. S. Fuhrmann, S.O. Krumke, H.-C. Wirth, Multiple hotlink assignment. In: A. Brandstadt, V.B. Le (Eds.), *WG 2001*, LNCS 2204, Springer, (2001) 189–200.
70. T. Fujie, An exact algorithm for the maximum leaf spanning tree problem. *Computers & Oper. Res.* 30(13) (2003) 1931–1944.
71. G.W. Furnas, J. Zacks, Multitrees: Enriching and reusing hierarchical structure. In: *Proc. of Int. Conf. Human Factors in Computing Systems CHI'94*, Boston, (1994) 330–336.
72. H.W. Gabow, Z. Galil, T. Spencer, R.E. Tarjan, Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica* 6(2) (1986) 109–122.
73. M. Gaeta, F. Orciuoli, S. Paolozzi, S. Salemo, Ontology extraction for knowledge reuse: The e-learning perspective. *IEEE Trans. SMC, Part C* 41(4) (2011) 798–809.
74. A. Gangemi, Ontology design patterns for semantic Web content. In: Y. Gal et al. (Eds.), *ISWC 2005*, LNCS 3729, Springer, (2005) 262–276.
75. M.R. Garey, D.S. Johnson, *Computers and Intractability. The Guide to the Theory of NP-Completeness*, San Francisco: W.H. Freeman and Company, 1979.
76. Gavish B., Topological design of telecommunication networks - the overall design problem, *EJOR* 58(2) (1992) 149–172.
77. L. Georgiadis, R.T. Tarjan, Dominator tree certification and independent spanning trees. Electronic preprint, 44 pp., Oct. 31, 2012. <http://arxiv.org/abs/1210.8303> [cs.DS]
78. J. Gehrke, Scalable decision tree construction. In: L. Liu, M.T. Ozsu Eds.), *Encyclopedia of Database Systems 2009*, New York, Springer, (2009) 2400–2469.
79. J. Gehrke, R. Ramakrishnan, V. Ganti, RainForestA framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery* 4(2/3) (2000) 127–162.
80. B. Gfeller, Faster swap edge computation in minimum diameter spanning trees. *Algorithmica* 62(1-2) (2012) 169–191.
81. I. Godor, G. Magyar, Cost-optimal topology planning of hierarchical access networks. *Computers and Operations Research* 32(1) (2005) 59–86.
82. M.X. Goemans, Y.-s. Myung, A catalog of Steiner tree formulations. *Networks* 23(1) (1993) 19–28.
83. R.M. Goodman, P. Smyth, Decision tree design from a communication theory standpoint. *IEEE Trans. Information Theory* 34(5) (1988) 979–994.
84. R.M. Goodman, P.A. Smyth, Decision tree design using information theory. *Knowledge Acquisition*

- 2(1) (1990) 1–19.
85. A.D. Gordon, *Classification*. 2nd ed., Chapman & Hall/CRC, London/Boca Raton, FL, 1999.
  86. T.R. Gruber, A translation approach to portable ontology specification. *Knowledge Acquisition* 5(2) (1993) 199–220.
  87. M.V. Gubko, The search for optimal organizational hierarchies with homogeneous manager cost functions. *Autom. & Remote Control* 69(1) (2008) 89–104.
  88. S. Guha, S. Khuller, Approximation algorithms for connected dominating sets. *Algorithmica* 20(4) (1998) 179–193.
  89. S. Guha, S. Khuller, Improved methods for approximating node weighted Steiner trees and connected dominating sets. *Information & Computation* 150(1) (1999) 57–74.
  90. A. Gupta, M. Pal, Stochastic Steiner tree without root. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M., Eds., *Proc. of 32nd Int. Colloquium on Automata, Languages and Programming ICALP 2005* LNCS 3580, Springer, (2005) 1051–1063.
  91. A. Gupta, M. Hajiaghayi, A. Kumar, Stochastic Steiner tree with non-uniform inflation. In: M. Charikar, K. Jansen, O. Reingold, J.D.P. Rolim, (Eds.), *Approx., Randomization, and Comb. Optim.: Algorithms and Techniques, APPROX-RANDOM 2007*, LNCS 4627, Springer, (2007) 134–148.
  92. S. Gutner, Elementary approximation algorithms for prize collecting Steiner tree problems. in: Yang, B., Du, D.-Z., Wang, C.A., Eds., *2nd Annu. Int. Conf. on Combinatorial Optimization and Applications COCOA 2008*, LNCS 5165, Springer, (2008) 246–254.
  93. J.-W. Ha, B.-H. Kim, B.-T. Zhang, Layered hypernetwork models for cross-modal associative text and image keyword generation in multimodal information retrieval. In: B.-T. Zhang, M.A. Orgun (Eds.), *PRICAI 2010*, LNCS 6230, Springer, (2010) 76–87.
  94. Y.Y. Haimes, Hierarchical holographic modeling. *IEEE Trans. SMC* 11(9) (1981) 606–617.
  95. Y.Y. Haimes, K. Tarvainen, T. Shima, J. Thadathil, *Hierarchical Multiobjective Analysis of Large-Scale Systems*. Hemisphere Pub. Corp., New York, 1990.
  96. H.W. Hamacher, G. Ruhe, On spanning tree problem with multiple objectives. *Ann. of Oper. Res.* 52(4) (1995) 209–230.
  97. D. Harel, STATECHARTS: A visual formalism for complex systems, *Science of Computer Programming* 8(3) (1987) 231–274.
  98. H. Holler, S. Voss, Software tools for multilayer network design. Presentation, *IV Int. Conf. on Decision Support for Telecommunications and Information Society*, Warsaw, Poland, Sept. (2004).
  99. C.W. Holsapple, K.D. Joshi, A collaborative approach to ontology design. *Commun. of the ACM* 45(2) (2002) 42–47.
  100. F. Hoppner, F. Klawonn, R. Kruse, T. Runkler, *Fuzzy Cluster Analysis*, New York: Wiley, 1999.
  101. F.K. Hwang, D.S. Richards, P. Winter, *The Steiner Tree Problem*. Amst.: Elsevier, 1992.
  102. M. Imase, B.M. Waxman, Dynamic Steiner tree problem. *SIAM J. on Discr. Math.* 4(3) (1991) 369–384.
  103. A. Itai, M. Rodeh, The multitree approach to reliability in distributed networks. *Information and Computation* 79(1) (1984) 43–59.
  104. T. Jacobs, On the complexity of optimal hotlink assignment. *Algorithmica* 62(3-4) (2012) 982–1005.
  105. J. Johnson, P. Irvani, The multilevel hypernetwork dynamics of complex systems of robot soccer agents. *ACM Trans. on Autonomous and Adaptive Systems* 2(2) (2007) Art. 5.
  106. I. Jonyer, D.J. Cook, L.B. Holder, Graph-based hierarchical conceptual clustering. *J. of Machine Learning Res.* 2 (2001) 19–43.
  107. R. Joobbani, D.P. Siewiorek, WEAVER: A knowledge-based routing expert. *IEEE Design and Test of Computers* 3(1) (1986) 12–23.
  108. A. Kapsalis, V.J. Rayward-Smith, G.D. Smith, Solving the graphical Steiner tree problem using genetic algorithm. *J. of the ORS* 44(4) (1993) 397–406.
  109. P. Kazienko, P. Brodka, K. Musial, J. Gaworecki, Multi-layer social network creation based on bibliographic data. In: A.K. Elmagarmid, D. Agrawal (Eds.), *SocialCom/PASSAT*, IEEE Computer Society, (2010) 407–412.
  110. H. Kellerer, U. Pferschy, D. Pisinger, *Knapsack Problems*. Berlin, Springer, 2004.
  111. S. Khuller, Approximation algorithms for finding highly connected subgraphs. In: D.S. Hochbaum (Ed.), *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, Boston, (1997)



- 236–265.
- 112.S. Khuller, A. Zhu, The general Steiner star problem. *Inf. Proc. Lett.* 84(4) (2002) 215–220.
- 113.P. Klein, R. Ravi, A nearly best-possible approximation algorithm for node-weighted Steiner trees. *J. of Algorithms* 19(1) (1995) 104–115.
- 114.D. Kleitman, D. West, Spanning trees with many leaves. *SIAM J. on Discr. Math.* 4(1) (1991) 99–106.
- 115.D.E. Knuth, *The Art of Computer Programming*. vol. 1, 3rd ed., vol. 2, 3rd ed., vol. 3, 2nd ed., Mass: Addison Wesley, Reading. 1997, 1997, 1998.
- 116.E. Kranakis, D. Krizanc, S. Shende, Approximate hotlink assignment. In: *Proc. of 12th Ann. Int. Symp. on Algorithms and Computation*, LNCS 2223, Springer, (2001) 756–767.
- 117.E. Kranakis, D. Krizanc, S. Shende, Approximate hotlink assignment. *Inf. Process. Lett.* 90(3) (2004) 121–128.
- 118.N.A. Kuznetsov, M.Sh. Levin, V.M. Vishnevsky, Some Combinatorial Optimization Schemes for Multi-Layer Network Topology. *Electronic Proc. of 17th IMACS World Congress 2005*, Paris, France, (2005) Paper T4-I-42-0486.
- 119.E. Laber, Hotlink assignment on the Web. In: R. Ahlswede et al, (Eds.), *Information Transfer and Combinatorics*, LNCS 4123, Springer, (2006) 1088–1092.
- 120.S. Levachkine, A. Guzman-Arenas, Hierarchy as a new data type for quantitative variables. *Expert Systems with Applications* 32(3) (2007) 899–910.
- 121.M.Sh. Levin, An extremal problem of organization of data. *Eng. Cybern.* 19(5) (1981) 87–95.
- 122.M.Sh. Levin, A hierarchical hypertext system. *Automatic Documentation and Math. Linguistics*, 23(3) (1989) 52–59.
- 123.M.Sh. Levin, *Combinatorial Engineering of Decomposable Systems*. Kluwer, Dordrecht, 1998.
- 124.M.Sh. Levin, *Composite Systems Decisions*, Springer, London, 2006.
- 125.M.Sh. Levin, Towards hierarchical clustering. In: V. Diekert, M. Volkov, A. Voronkov, (Eds.), *CSR 2007*, LNCS 4649, Springer, (2007) 205–215.
- 126.M.Sh. Levin, Combinatorial optimization in system configuration design. *Automation and Remote Control* 70(3) (2009) 519–561.
- 127.M.Sh. Levin, Towards communication network development (structural systems issues, combinatorial problems). *IEEE Region 8 Int. Conf. Sibircon 2010*, vol. 1, (2010) 204–208.
- 128.M.Sh. Levin, Aggregation of composite solutions: strategies, models, examples. Electronic preprint, 72 pp., Nov. 29, 2011. <http://arxiv.org/abs/1111.6983> [cs.SE]
- 129.M.Sh. Levin, Restructuring in combinatorial optimization. Electronic preprint. 11 pp., Febr. 8, 2011. <http://arxiv.org/abs/1102.1745> [cs.DS]
- 130.M.Sh. Levin, Four-layer framework for combinatorial optimization problems domain. *Advances in Engineering Software* 42(12) (2011) 1089–1098.
- 131.M.Sh. Levin, Morphological methods for design of modular systems (a survey). Electronic preprint, 20 pp., Jan. 9, 2011. <http://arxiv.org/abs/1201.1712> [cs.SE]
- 132.M.Sh. Levin, Towards electronic shopping of composite product. Electronic Preprint. 10 pp., March 3, 2012. <http://arxiv.org/abs/1203.0648> [cs.SE]
- 133.M.Sh. Levin, Multiset estimates and combinatorial synthesis. Electronic preprint. 30 pp., May 9, 2012. <http://arxiv.org/abs/1205.2046> [cs.SY]
- 134.M.Sh. Levin, Composite strategy for multicriteria ranking/sorting (methodological issues, examples). Electronic preprint. 24 pp., Nov. 9, 2012. <http://arxiv.org/abs/1211.2245> [math.OC]
- 135.M.Sh. Levin, M.L. Nisnevich, Combinatorial scheme for management of life cycle: example for concrete macrotechnology. *J. of Intell. Manuf.* 12(4) (2001) 393–401.
- 136.M.Sh. Levin, L.V. Sokolova, Hierarchical combinatorial planning of medical treatment. *Computer Methods and Programs in Biomedicine* 73(1) (2004) 3–11.
- 137.M.Sh. Levin, M.A. Danieli, Hierarchical Morphological Decision Making Framework for Evaluation and Improvement of Composite Systems (Example for Building) *Informatica* 16(2) (2005) 213–240.
- 138.M.Sh. Levin, M.V. Petukhov, Multicriteria assignment problem (selection of access points). *Proc. of 23rd Int. Conf. IEA/AIE 2010, "Trends in Applied Intelligent Systems"*, LNCS 6097, part II, Springer, (2010) 277–287.
- 139.M.Sh. Levin, A.V. Safonov, Towards modular redesign of networked system, *Proc. of Int. Conf. on Ultra Modern Telecommunication 'ICUMT 2010'*, Moscow, (2010) 109–114.

- 140.M.Sh. Levin, R.I. Nuriakhmetov, Multicriteria Steiner tree problem for communication network. Electronic preprint, 11 pp., Febr. 12, 2011. <http://arxiv.org/abs/1102.2524> [cs.DS]
- 141.M.Sh. Levin, A.A. Zamkovoy, Multicriteria Steier tree wit the cost of Steiner vertices. *J. of Communications Technology and Electronics* 56(12) (2001) 1527-1542.
- 142.W. Liang, Constructing minimum-energy broadcast trees in ad hoc wireless networks. In: *Proc. of 3rd MOBIHOC*, ACM, New York, (2002) 112-122.
- 143.G.-H. Lin, G. Xue, Steiner tree problem with minimum number of Steiner points and bounded edge-length. *Information Processing Letters* 69(2) (1999) 53-57.
- 144.G.-H. Lin, G. Xue, On the terminal Steiner tree problem. *Information Processing Letters* 84(2) (2000) 103-107.
- 145.J. Lin, Integration of weighted knowledge bases. *Artificial Intelligence* 83(2) (1996) 363-378.
- 146.I. Ljubic, R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, M. Fischetti, An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Math. Progr. Ser. B* 105(2-3) (2006) 427-449.
- 147.M.F. Lopez, A. Gomez-Perez, J.P. Sierra, A.P. Sierra, Building a chemical ontology using methodology and the ontology design environment. *IEEE Intelligent Systems* 14(1) (1999) 37-46.
- 148.H. Lu, R. Ravi, The power of local optimization: Approximation algorithms for maximum leaf spanning tree. In: *Proc. of the Annual Allerton Conf. on Communication Control and Computing*, USA, Univ. of Illinois, vol. 30, (1992) 533-542.
- 149.H. Lu, R. Ravi, Approximating maximum leaf spanning trees in almost linear time. *J. of Algorithms* 29(1) (1998) 132-141.
- 150.T. Lust, J. Teghem, The multiobjective multidimensional knapsack problem: a survey and a new approach. Techn. Rep., arXiv: 1007.4063v1, 2010. <http://arxiv.org/abs/1007.4063.v1>
- 151.T.L. Magnanti, L.A. Wolsey, Optimal trees. In: M.O. Ball et al. (Eds.), *Handbooks on OR & MS*, North-Holland, Amsterdam, vol. 7, (1995) 503-615.
- 152.M. Magnani, L. Rossi, The ml-model for multi-layer social networks. In: ASONAM, IEEE Computer Society, (2011) 5-12.
- 153.M. Magnani, L. Rossi, Multi-stratum networks: toward a unified model of on-line identities. Electronic preprint, 18 pp., Nov. 1, 2012. <http://arxiv.org/abs/1211.0169> [cs.SI]
- 154.J.G. March, H.A. Simon, *Organizations*. 2nd ed., Wiley-Blackwell, 1993.
- 155.V. Marianov, D. Serra, Hierarchical location-allocation models for congested systems. *EJOR* 135(1) (2001) 196-209.
- 156.S. Martello, P. Toth, *Knapsack Problem: Algorithms and Computer Implementation*, New York: Wiley, 1990.
- 157.R. Matichin, D. Peleg, Approximation algorithm for hotlink assignment in the greedy model. *Theor. Comput. Sci.* 383(1) (2004) 102-110.
- 158.M.D. Mesarovic, D. Macko, Y. Takahara, *Theory of Hierarchical, Multilevel Systems*. New York and London: Academic Press, 1970.
- 159.B. Mirkin, F.R. McMorris, F.S. Roberts, A. Rzhetsky, (Eds.), *Mathematical Hierarchies and Biology*, DIMACS Ser. in Discrete Math. and Theoretical Comput. Sci., Providence: AMS, 1996.
- 160.S. Mishin, Optimal organizational hierarchies in firms. *J. of Business Economics and Management* 8(2) (2007) 79-99.
- 161.M. Molnar, A. Bellabas, S. Lahoud, The cost optimal solution of the multi-constrained multicast routing problem. *Computer Networks* 56(13) (2012) 3136-3149.
- 162.M.W. Murhammer, K.-K. Lee, P. Motallebi, P. Borghi, K. Wozabal, *IP Network Design Guide*, IBM Red Book, 1999.
- 163.M.A. Mussen, Dimensions of knowledge sharing and reuse. *Computers and Biomedical Research* 25 (1992) 435-467.
- 164.N.F. Noy, C. Hafner, The state of the art in ontology design: A survey and comparative review. *AI Magazine* 18(3) (1997) 53-74.
- 165.N.F. Noy, M.A. Musen, PROMPT: Algorithm and tool for automated ontology merging and alignment. AAAI, 2000.
- 166.N. Noy, A.H. Doan, A.Y. Halevy, Semantic integration. *AI Magazine* 26(1) (2005) 7-10.
- 167.C. Obreque, V. Marianov, M. Rios, Optimal design of hierarchical network with free main path

- extremes. *Operations Research Lett.* 36(3) (2008) 366–371.
- 168.C. Obreque, M. Donoso, G. Gutierrez, V. Marianov, A branch and cut algorithm for the hierarchical network design problem. *EJOR* 200(1) (2010) 28–35.
- 169.V. Pareto, *Manual of Political Economy*. (English translation), A. M. Kelley Publishers, New York, 1971.
- 170.R. Parra-Hernandez, N. Dimopoulos, A new heuristic for solving the multichoice multidimensional knapsack problem. *IEEE Trans. SMC, Part A* 35(5) (2002) 708–717.
- 171.M. Perkowitz, O. Etzioni, Towards adaptive web sites: conceptual framework and case study. *Computer Networks* 31(11–16) (1999) 1245–1258.
- 172.S. Pettie, V. Ramachandran, An optimal minimum spanning tree algorithm. *J. of the ACM* 49(1) (2002) 16–34.
- 173.S. Pettie, V. Ramachandran, A randomized time-work optimal parallel algorithm for finding a minimum spanning forest. *SIAM J. on Computing* 31(6) (2002) 1876–1895.
- 174.H.S. Pinto, J.P. Martins, A methodology for ontology integration. In: *Proc. of 1st Int. Conf. Knowledge Capture K-CAP'01*, Victoria, Canada, Sweden, (2001) 113–138.
- 175.H. Pirkul, J. Current, V. Nagarajan, The hierarchical network design problem: A new formulation and solution procedures. *Transportation Science* 25(3) (1991) 175–182.
- 176.J.R. Quinlan, Induction of decision trees. *Machine Learning* 1(1) (1986) 81–106.
- 177.J.R. Quinlan, Decision trees and decision making. *IEEE Trans. SMC* 20(2) (1990) 339–346.
- 178.J. Rasmussen, The role of hierarchical knowledge representation in decision making and system management. *IEEE Trans. SMC* 15(2) (1985) 234–243.
- 179.D.F. Robinson, L.R. Foulds, Comparison of phylogenetic trees. *Mathematical Biosciences* 53(1/2) (1981) 131–147.
- 180.M.B. Rosenwein, R.T. Wong, Constrained Steiner tree problem. *EJOR* 81(2) (1995) 430–439.
- 181.B. Roy, *Multicriteria Methodology for Decision Aiding*. Kluwer, Dordrecht, 1996.
- 182.T.L. Saaty, *The Analytic Hierarchy Process*. New York, MacGraw-Hill, 1988.
- 183.S. Sahni, Approximate algorithms for 0/1 knapsack problem. *J. of the ACM* 22(1) (1975) 115–124.
- 184.S. Sahni, E. Horowitz, Combinatorial problems: reducibility and approximation. *Oper. Res.* 26(5) (1978) 718–759.
- 185.N. Saitou, T. Imanishi, Relative efficiencies of the Fitch-Margoliash, maximum-parsimony, minimum-likelihood, minimum-evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree. *Mol. Biol. Evol.* 6(5) (1989) 514–525.
- 186.L.A. Sanchis, Experimental analysis of heuristic algorithms for the dominating set problem. *Algorithmica* 33(1) (2002) 3–18.
- 187.M. Santos, L.M.A. Drummond, E. Uchoa, A distributed primal-dual heuristic for Steiner problems in networks. in: Demetrescu, C., Ed., *Proc. of 6th Int. Workshop on Exp. Algorithms WEA 2007*, LNCS 4525, Springer, (2007) 175–188.
- 188.K. Shiimoto, T. Kurimoto, GMPLS-based traffic engineering in multi-region/multi-layer service network. *MPLS Int. Conf.*, (2004).
- 189.A. Segev, The node-weighted Steiner tree problem. *Networks* 17(1) (1987) 1–17.
- 190.A. Singh, An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem. *Applied Soft Computing* 9(2) (2009) 625–631.
- 191.P.H.A. Sneath, R.R. Sokol, *Numerical Taxonomy: The Principles and Practice of Numerical Classification*. W.H. Freeman and Company, San Francisco, 1973.
- 192.R. Solis-Oba, 2-approximation algorithm for finding a spanning tree with maximum number of leaves. In: G. Bilardi, G.F. Italiano, A. Pietracaprina, G. Pucci (Eds.), *Proc. of 6th Annual Eur. Symp. on Algorithms - ESA '98*, LNCS 1461, Springer, (1998) 441–452.
- 193.J.A. Storer Constructing full spanning trees for cubic graphs. *Infor. Proc. Lett.* 13(1) (1981) 8–11.
- 194.G.J. Szekely, M.L. Rizzo, Hierarchical clustering via joint between-within distances: Extending Ward's minimum variance method. *J. of Classification* 22(2) (2005) 151–183.
- 195.A.S. Tanenbaum, *Computer Networks*. 4th ed., Prentice Hall PTR, NJ, 2002.
- 196.A.S. Tanenbaum, *Structured Computer Organization*, 5th ed., Prentice Hall, NJ, 2006.
- 197.A.S. Tannenbaum, B. Kavcic, M. Rosner, M. Vianello, G. Weisner, *Hierarchy in Organization*, Jossey-Bass, Inc., San Fransisco, 1974.

- 198.R.E. Tarian, Edge-disjoint spanning trees and depth-first search. *Acta Informatica* 6(2) (1976) 171–185.
- 199.M. Thai, F. Wang, D. Liu, S. Zhu, D. Du, Connected dominating sets in wireless networks with different transmission ranges. *IEEE Trans. Mob. Comput.* 6(7) (2007) 721–730.
- 200.E. Uchoa, Reduction tests for the prize-collecting Steiner problem. *Oper. Res. Lett.* 34(4) (2006) 437–444.
- 201.M. Uschold, M. Gruninger, Ontologies: Principles, methods and applications. *Knowledge Eng. Rev.* 11(2) (1996) 93–136.
- 202.A.A. Voronin, S.P. Mishin, Algorithms to seek the optimal structure of the organizational system. *Autom. & Remote Control* 63(5) (2002) 803–814.
- 203.S. Voss, Steiner’s problem in graphs: Heuristic methods. *Discr. Appl. Math.* 40(1) (1992) 45–72.
- 204.S. Voss, The Steiner tree problem with hop constraints. *Ann. of Oper. Res.* 86 (1999) 321–345.
- 205.S. Voss, Modern heuristic search methods for the Steiner tree problem in graph. in: Du, D.-Z., Smith, J.M., Rubinstein, J.H., Eds., *Advances in Steiner Trees*, Boston, Kluwer, (2000) 283–323.
- 206.M. Vujosevic, M. Stanojevic, A bicriteria Steiner tree problem on graph. *Yugoslav J. of Operations Research* 13(1) (2003) 25–33.
- 207.H. Wache, T. Vogele, U. Visser, H. Stickenschmidt, G. Schuster, H. Neumann, S. Hubner, Ontology-based integration of information - A survey of existing methods. In: *Proc. of Int. Workshop Ontologies and Information Sharing IJCAI-01*, Seattle, WA, (2001) 108–117.
- 208.H.J. Waller, The synthesis of hierarchical structures: Techniques and applications. *Decision Sciences* 7(4) (1976) 659–674.
- 209.J. Westbrook, D.C.K. Yan, Greedy algorithm for the on-line Steiner tree and generalized Steiner problems. In: Dehne, F.K.H.A., Sack, J.-R., Santoro, N., Whitesides, S., Eds., *Proc. of the Third Workshop on Algorithms & Data Structures WADS’93*, LNCS 709, Springer, (1993) 622–633.
- 210.P. Winter, Steiner problem in networks: A survey. *Networks* 17(2) (1987) 129–167.
- 211.B.Y. Wu, K. Chao, *Spanning Trees and Optimizaiton Problems*. CRC Press, Boca Raton, FL, 2003.
- 212.G. Wynants, *Network Synthesis Problems*, Dordrecht, Kluwer, 2001.
- 213.R.R. Yager, On ordered weighted averaging operators in multicriteria decision making. *IEEE Trans. SMC* 18(1) (1988) 183–190.
- 214.L. Yelowitz, An efficient algorithm for constructing hierarchical graphs. *IEEE Trans. SMC* 6(4) (1976) 327–329.
- 215.H. Zimmerman, OSI Reference Model - The ISO model of architecture for open systems interconnection. *IEEE Trans. on Communications* 28(4) (1980) 425–432.
- 216.F. Zou, X. Li, D. Kim, W. Wu, Two constant approximation algorithms for node-weighted Steiner tree in unit disk graphs. in: Yang, B., Du, D.-Z., Wang, C.A. (Eds.), *Proc. of Second Annu. Int. Conf. on Comb. Optim. and Appl. COCOA 2008*, LNCS 5165, Springer, (2008) 278–285.
- 217.F. Zwicky, *Discovery Invention, Research Through the Morphological Approach*. McMillan, New York, 1969.